UCLA-ENG-7167
October 1971

ARPA CONTRACT NO. DAHC-15-69-C-0285

NODAL BLOCKING IN LARGE NETWORKS

Jack F. Zeigler

COMPUTER SYSTEMS
MODELING AND ANALYSIS GROUP

COMPUTER SCIENCE DEPARTMENT

School of Engineering and Applied Science
University of California
Los Angeles

160

COMPUTER SYSTEMS MODELING AND ANALYSIS GROUP
REPORT SERIES

[illegible]

published by

REPORTS GROUP
SCHOOL OF ENGINEERING AND APPLIED SCIENCE
UNIVERSITY OF CALIFORNIA, LOS ANGELES 90024

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) University of California School of Engineering and Applied Science Los Angeles, California 90024	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED
	2b. GROUP

3. REPORT TITLE
NODAL BLOCKING IN LARGE NETWORKS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

5. AUTHOR(S) (First name, middle initial, last name)
Jack F. Zeigler

6. REPORT DATE October 1971	7a. TOTAL NO. OF PAGES 161	7b. NO. OF REFS 27

8a. CONTRACT OR GRANT NO. DAHC-15-69-C-0285	9a. ORIGINATOR'S REPORT NUMBER(S) UCLA-ENG-7166
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

A theoreticel study is given for store-and-forwerd communication networks in which the nodes have finite storage cepacity for messages. A node is "blocked" when its storege is filled, otherwise it is "free." A two-state Markov model is proposed for each node, and the frection of blocked nodes in the network is shown also to have a two-state Markov process represententation. The time-dependent probability thet any given node in the network is blocked is obtained for some uniform networks of arbitrary dimension, and verious results describe the clumping phenomena in these networks.

Through e modification of the basic Markovian network model, the fraction of blocked nodes in a computer-simuleted store-end-forward communication network is predicted with reasoneble accurecy.

DD FORM 1 NOV 65 1473 (PAGE 1)
0102-014-6600

UNCLASSIFIED
Security Classification

UNCLASSIFIED
Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Networks						
Computer Networks						
Approximation for Network Blocking						
Blocking in Networks						
Storage Blocking						
ARPA Computer Networks						

DD FORM 1 NOV 65 1473 (BACK)
(PAGE 2)

UNCLASSIFIED
Security Classification

UCLA-ENG-7167

NODAL BLOCKING IN LARGE NETWORKS

by

Jack F. Zeigler

Supported by

ADVANCED RESEARCH PROJECTS AGENCY
Computer Network Research
ARPA Order No. 1380, Program Code No. 9D30
Contract No. DAHC-15-69-C-0285

Leonard Kleinrock, Principal Investigator

Computer Systems Modeling and Analysis Group

Computer Science Department
School of Engineering and Applied Science
University of California
Los Angeles, California 90024

October 1971

PREFACE

The research described in this report, "Nodal Blocking in Large Networks," by Jack Zeigler, is part of a continuing investigation of Computer Network Research, sponsored by the Advanced Research Projects Agency (ARPA), Department of Defense Contract DAHC-15-69-C-0285, under the direction of L. Kleinrock, Principal Investigator, and G. Estrin, M. Melkanoff, and R. Muntz, Co-Principal Investigators, in the Computer Science Department of the School of Engineering and Applied Science, University of California, Los Angeles. This project was also partially sponsored by a National Science Foundation Traineeship.

This report was the basis of a Ph.D. dissertation (June 1971) submitted by the author under the chairmanship of Leonard Kleinrock.

ACKNOWLEDGEMENTS

I would like to thank the members of my committee, Professors David G. Cantor, Wesley W. Chu, James R. Jackson, Leonard Kleinrock (Chairman), and Richard R. Muntz, for their assistance in this work. Special thanks go to Dr. Cantor, who helped me obtain the solutions given in Appendix A, and to my advisor, Dr. Kleinrock, who introduced me to the nodal blocking problem and cleverly guided my work. Thanks also go to Frank Kontrovich and Charlotte LaRoche, who aided me with their sound judgment.

This research was supported in part by a National Science Foundation Traineeship and the Advanced Research Projects Agency of the Department of Defense under Contract #DAHC-15-69-C-0285.

ABSTRACT

A theoretical study is given for store-and-forward communication networks in which the nodes have finite storage capacity for messages. A node is "blocked" when its storage is filled, otherwise it is "free." A two-state Markov model is proposed for each node, and the fraction of blocked nodes in the network is shown also to have a two-state Markov process representation. The time-dependent probability that any given node in the network is blocked is obtained for some uniform networks of arbitrary dimension, and various results describe the clumping phenomena in these networks.

Through a modification of the basic Markovian network model, the fraction of blocked nodes in a computer-simulated store-and-forward communication network is predicted with reasonable accuracy.

CONTENTS

CONTENTS (Continued)

FIGURES

CHAPTER 1

INTRODUCTION

A. Computer Networks

In the early 1960's the first time-sharing facility began operation. Since that time, facilities and systems have grown and developed across the country into quite sophisticated and unique sites each having special features and capabilities in the form of exceptional computer programs, data files, hardware devices, resources, and human talent which, in general, are not easily transferable. A desire to share these resources has led to the development of computer networks which permit the separate computer facilities to communicate with each other.

A computer network is a collection of nodes (computers) connected together by a set of links or lines (communication channels). Messages in the form of commands, inquiries, replies, and file transmissions travel through this network over data transmission lines. At the nodes, the task of relaying messages (with all proper routing, acknowledging, error control, queueing, etc.) and inserting and removing messages which originate and terminate at that node must be carried out.

The Advanced Research Projects Agency (ARPA) Network [1-5] is a store-and-forward computer communication network linking approximately fifteen research centers across the country at the present time with approximately five more scheduled for completion by the end of 1971. In a store-and-forward network, messages are broken up into convenient sized packets that individually make their way through the net, "hopping" from node to node. If a packet cannot be transmitted immediately out of a node on its way through the net because its designated output line is in use, it forms a queue and awaits its turn to be transmitted.

Western Union has used the store-and-forward concept for years as has the United States Air Force in its Sage Defense System. In November of 1969 DATRAN Corporation proposed to the Federal Communications Commission a network for digital communication linking 35 metropolitan areas from Boston to San Francisco and comprising 240 microwave relay stations. Eventually they propose to make it a store-and-forward network [6]. We see that numerous store-and-forward networks are already in use and others are being planned.

B. Structure of the ARPA Network

Let us examine the structure of the ARPA Network more carefully. At each site in the network there is at least one large digital computer called a HOST, which acts as a source and terminal for messages in the network. These computers are basically incompatible in hardware, software, file structure, etc., and hence there is a need for an intermediate device to interface these HOSTs to the communication net which connects them. This function, and others, is performed in the ARPA Network at each site by a digital computer called an Interface Message Processor (IMP). The IMPs carry out the message handling process in the network, so when we speak of the nodes in the net we are actually referring to the various IMPs.

An IMP in the ARPA net receives messages from two sources:

1. Other IMPs like itself over fully duplex 50 Kbit/sec. leased telephone lines.

2. One or more HOSTs over 100 Kbit/sec. fully duplex lines. Message bits are sent in series and are protected by error detection schemes. If an error is detected, the message must be retransmitted.

Compared to any HOST computer, the IMP is a small machine with finite storage space for messages. Part of the IMP storage is strictly allocated for messages which are relayed from neighboring IMPs and which must be transmitted to still another IMP before reaching their destination; this is called store-and-forward traffic. Part of the remaining storage in an IMP is strictly allocated for the reassembly of multi-packet messages destined for one of the IMP's HOSTs. (A multi-packet message is one which is too large to be transmitted as a single packet whose maximum size is 1008 bits. Multi-packet messages may be up to 8 packets in length, and each of these packets must be held until all are received in the final node, at which time they are reassembled into the original message and delivered to the HOST. Longer messages must be partitioned in the HOST into many multi-packet messages.) The remaining storage is allocated between these two types of traffic as needed. In all, the IMP contains storage space for about 50 single-packet messages.

C. Nodal Blocking

From time to time, during periods of high utilization, the IMP's storage can become filled, so that arriving messages must be refused. When this occurs we say that the node is "blocked." Blocking in the IMP can occur in any of three ways:

1. There are no more reassembly spaces available for HOST traffic, and packets for a HOST that were sent by other IMPs must be refused.

2. There are no more spaces available for store-and-forward

traffic, and thus non-HOST packets must be refused.

3. There are no more spaces for arriving messages and all traffic must be refused.

Certain high priority messages are never blocked, e.g., space is always saved for positive acknowledgments sent by neighboring IMPs to indicate that a message previously sent by the IMP has been received without error and can thus be discarded by the IMP. On the other hand, a blocked message is ignored by the IMP, and the absence of a positive acknowledgment tells the IMP which sent the message that the message will require retransmission.

Selective blocking, as in points (1) and (2) above, or total blocking, as in (3), can occur in this network if the input rate of messages equals or exceeds the output capacity over a period of time. We would normally expect this to occur only during peak hours of the day. However, it is a potentially dangerous situation because a blocked neighbor reduces a node's message output rate with no corresponding change in its input rate. This causes its storage to fill at a faster rate and increases its chance of becoming blocked. Thus blocking could propagate in both space and time.

The purpose of this research is to gain an understanding of the blocking behavior in a message-switching network.

CHAPTER 2

THE MODEL

A. General Description

Selective blocking is a very difficult problem to analyze. The allocation of storage between store-and-forward traffic and HOST traffic is equivalent to the formation of two distinct queues with finite waiting room, or storage space, in which the maximum size of the waiting room for each queue is dependent on the number of customers (i.e. messages) in the other queue. To make the problem mathematically tractable, the network we analyze will consist of nodes having a single queue for messages. If there is an empty space in the queue, the first arriving message, regardless of its final destination, will take that space. If there are no spaces for arriving messages, then the node is "blocked."

As soon as one message is transmitted by a blocked node, it becomes a "free" node. It remains in this state as long as there is at least one empty space in storage that could be used by an arriving message. When the storage fills again, the node re-enters the blocked state.

Figure 1 shows a simplified model of such a node in the terminology of the ARPA Network. The IMP, when free, accepts messages into its main storage from two sources:

1. Other IMPs.
2. A single HOST which generates and receives messages (as a source and terminal).

A message in a message buffer is queued up for transmission over an

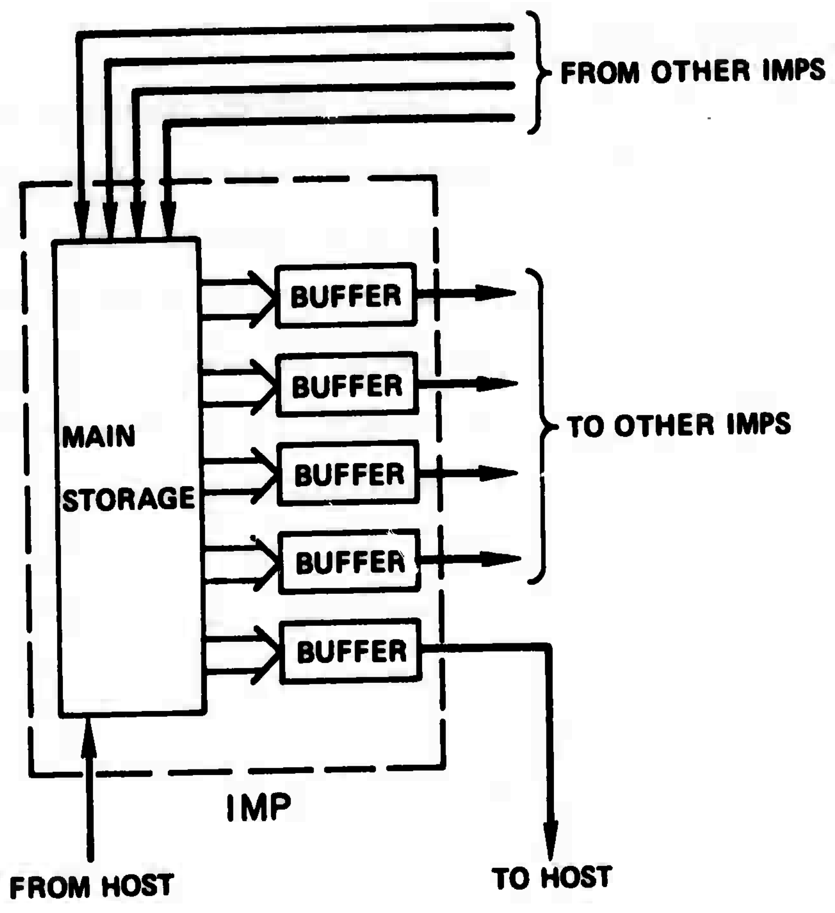


Figure 1. Schematic of a Node

appropriate output line to some neighbor as determined by the final destination of the message, and is then transmitted serially to that neighbor. Any of these neighbors can become blocked, thus preventing the use of the output line feeding such neighbors.

Nodal blocking is caused by the finite storage room for messages in the IMP and the overutilization of the system. By overutilization, we mean that when the node is accepting messages, its average arrival rate equals or exceeds its average service rate (which is the total output channel capacity divided by the average message length). Elementary queueing theory [7] shows that if (1) the system is underutilized, and (2) there is storage space for approximately twenty messages or more, then under fairly general conditions there will be essentially no blocking.

The analysis of the propagation of blocking is difficult for at least three reasons. First, it involves networks of queues for which only stationary results at best can generally be obtained. Second, the pertinent stochastic processes are dependent, for if a node becomes blocked, it cannot accept messages from its neighbors and their storage will tend to fill at a faster rate. Finally, it is a transient queueing problem and even the simplest of these is very difficult to solve. (For example, the queueing system with Markovian arrivals, a single exponential server, and unlimited waiting room has modified Bessel functions in its time dependent solution [7].)

B. Related Work

A number of topics in graph theory are related to this problem. Ignition phenomena as developed by Rapoport [8] and Allanson [9] treats vertices (nodes) which are "excited" if they receive a certain minimum number of stimuli within a certain amount of time. This excitation is assumed to stimulate d other vertices to which it is randomly connected. Stable states, i.e., constant fractions of vertices being excited, are shown to exist under some conditions. This model has immediate application to neural networks because of their essentially random connectivity and the nearly deterministic behavior of neurons. However, the model cannot be reasonably applied to computer networks because they are not randomly connected and the probabilistic nature of information transfer in the form of variable length and time of arrival of messages makes the excitation process (i.e. the blocking) very much non-deterministic.

Percolation Theory [10] considers lattices in which a branch between any two nodes is present with probability p or deleted with probability $1 - p$. The main concern here is the minimum value of p (i.e. the critical value) for which a connected component of infinite length exists in the lattice with probability one. The relation of this theory to the work of Gilbert [11] on random plane networks is clear.

In the study of probabilistic graphs [12], branches and/or vertices are deleted in some random fashion. The questions raised (and answered) are the following: what are the probabilities corresponding to various kinds of connectivity; what is the distribution of the size of the largest connected component, etc.

An interesting variation on the network vulnerability problem is that which considers a probabilistic repair time for vertices or branches that have been damaged by an attack from some weapon system. In [13] the time varying probability of connectivity is determined for random graphs.

In none of these areas of graph theory is the state of a node (or vertex) ever taken to be a function of the states of its neighbors. Thus such results are not applicable to the study of blocking propagation.

Eden [14] and Morgan and Welsh [15] studied two-dimensional Poisson growth processes. They assumed that "infection" in a cell network spread from cell to neighboring cell in an amount of time taken from some probability distribution. These authors obtained results on the shape of the infected area and the rate of spread of the infection. In their models, once a cell becomes infected it remains in that state forever, thus their work cannot be taken as a solution to the blocking problem.

Roach [16] studied the overlap of objects placed at random in some space and called these overlappings "clumps." He treats the number of clumps, their size, their shape, and the spacing between them for a number of interesting cases, including the square lattice. He assumes the probability that a lattice point is marked (i.e. blocked) is the same for all lattice points. Because of this independence assumption and also because his system is static, we cannot utilize his results; however, we will adopt his terminology.

C. The Mathematical Model

The blocking problem is a difficult one. Since we cannot solve the problem exactly, our goal is to make good approximations that allow us to analyze the system and characterize its blocking behavior in some way. To this end we make the following assumptions:

1. The HOST cannot become blocked (it is an infinite sink).

2. a. Input traffic from the HOST is Poisson.

   b. Traffic on all lines (including the HOST-IMP line) has the same average rate so that total traffic into each node is σ messages/sec.

3. a. Message lengths are exponentially distributed.

   b. Service (transmission) time on any line is therefore exponentially distributed such that for a node with k blocked neighbors, the rate at which messages exit from that node is $\mu^{(k)}$ messages/sec., dependent on the number of blocked neighbors.

4. The probability of an empty queue in the IMP is approximately zero (since the system is assumed to be overutilized.

CHAPTER 3

ANALYSIS

A. The Nodal Model

Under the assumptions in "The Mathematical Model" (Section 2.C), we arrive at a simplified blocking model for a node in the network as a two-state Markov process (Fig. 2). If the node is blocked, i.e., in state b, it becomes free in the next instant of time Δt with probability $\mu^{(k)}\Delta t$ where k is the number of blocked neighbors it is experiencing at that time. Similarly, if the node is free, i.e., in state f, it becomes blocked in the next instant of time Δt with probability $\lambda^{(k)}\Delta t$ where k is again the number of blocked neighbors. Thus $\lambda^{(k)}$ is the rate at which a free node becomes blocked in the presence of k blocked neighbors, and should increase with k. $\mu^{(k)}$, on the other hand, being the rate at which a blocked node becomes free, should decrease with k.

1. Derivation of $\mu^{(k)}$

Below we show the appropriateness of this model. First, we require the Laplace transform of the message interdeparture time probability density $\equiv D(s)$. For any node let $\rho \equiv P$[non-empty node] and let the Laplace transform of the probability density of the message interarrival time process be $A(s)$. Because we have assumed that the service time is exponential with parameter $\mu^{(k)}$, we know that the Laplace transform of the departure process, conditioned on a non-empty system is

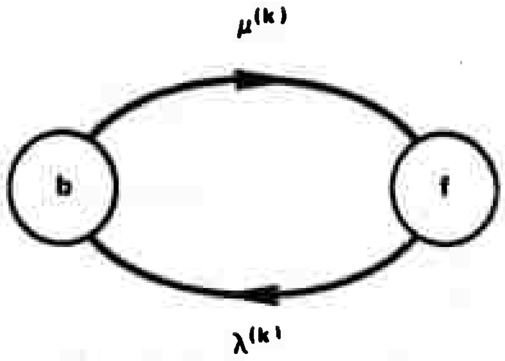
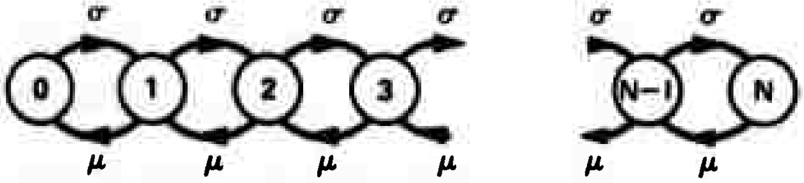
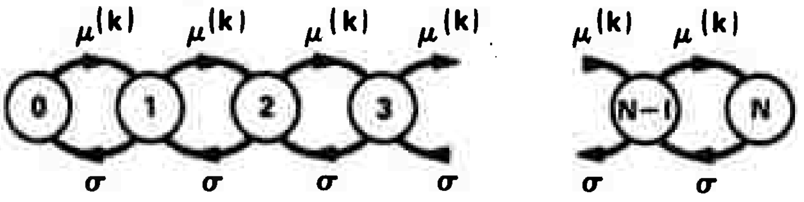


Figure 2. Blocking Model for an Imp



a) QUEUE STATE TRANSITIONS



b) DUAL QUEUE STATE TRANSITIONS

Figure 3

$\mu^{(k)}/(s + \mu^{(k)})$. Therefore,

$$D(s) = \frac{\rho\mu^{(k)}}{s + \mu^{(k)}} + (1 - \rho)\, A(s)\, \frac{\mu^{(k)}}{s + \mu^{(k)}} \tag{1}$$

By assumption (4) we have $\rho \approx 1$

$$D(s) \approx \frac{\mu^{(k)}}{s + \mu^{(k)}} \tag{2}$$

which says that the departure process is a Poisson stream. See Burke [17] and Reich [18] for further details on departure processes.

We have assumed that the traffic on all lines has the same average rate. If, for example, every node has exactly four neighbors and one HOST, then there are five output lines from each node. All of these lines are equivalent (except that the HOST cannot become blocked) and, by the assumption of exponential message lengths, the departure process from each output line constitutes a Poisson stream at rate $\mu^{(0)}/5$ when that neighbor is not blocked (and at rate 0 when that neighbor is blocked).

$$\mu^{(k)} = \frac{5 - k}{5}\, \mu^{(0)} \qquad k = 0,1,\ldots,4 \tag{3}$$

where $\mu^{(0)}$ is a given system parameter and represents the maximum message departure rate from a node. This set of numbers is merely an illustration; any combination can be treated by this model. These results show that we can approximate the time spent in the blocked state as being exponentially distributed with parameter $\mu^{(k)}$. Because of the memoryless property of the exponential distribution, the expected value of the remaining time to be spent in the blocked state, given that k changes to some new value k_n while in the blocked state, is simply $1/\mu^{(k_n)}$. By Eq. (3) this means that an increase in k should tend to increase the time spent in the blocked state, and a decrease in k should tend to decrease this time. We would expect to see such behavior in a real computer network.

2. Derivation of $\lambda^{(k)}$

The derivation of the parameter $\lambda^{(k)}$ is not nearly as simple. The time that an IMP spends in the free state is distributed as the busy period in a queueing system with finite queueing room for customers, as we now show. We begin by first considering the state transition diagram or Markov chain model for such a single node finite storage queueing system as shown in Figure 3a. The numbers inside the circles represent the number of customers (messages) in the node. We assume that customers arrive in a Poisson fashion with parameter σ, and depart after receiving service (exponentially distributed with an average of $1/\mu$ seconds). A busy period begins when a customer arrives to find an empty system (at which time he immediately enters the service facility). Customers arriving during his service time form a queue behind him. With each arrival the system moves to the right along the state transition diagram because the number in the system is increased by one, and with each service completion (i.e., departure) it moves to the left. Customers arriving when the system contains N customers are lost (i.e., depart without service). The busy period ends the first time the system goes empty after initiation of the busy period.

For the IMP model we now consider a dual queue in which the roles of service and arrival are reversed, and the numbers inside the circles now represent the number of empty places in storage that could be used by arriving messages (Fig. 3b). The free period of the IMP begins with the departure of a message from a previously filled system, i.e., no empty places for arriving messages. With a transmission (departure) the system moves from state 0 to state 1. It continues to move to the right with each transmission and to the left with each arrival. The free period ends the first time the system returns to the 0 state. The correspondence between the primal and dual queues is perfect; thus any results obtained for the busy period in the primal system are applicable to the dual queue free period in the IMP simply by substituting $\mu^{(k)}$ for σ and σ for μ, as in Figs. 3a,b.

The busy period for a finite queueing room system is difficult to obtain, but the result for unlimited queueing room is well known. The probability density of the length t of the busy period in such a system is

$$p^{(t)} = \frac{1}{t\sqrt{\rho}} e^{-(\sigma+\mu)t} I_1(2t\sqrt{\sigma\mu}) \tag{4}$$

where ρ, the utilization factor = $(\sigma/\mu) < 1$ and $I_1(x)$ is the modified Bessel function of the first kind, of order one [19]. If the size of the queueing room is greater than 20, the solution for unlimited queueing room is a good approximate solution to the limited queueing room problem. (This follows since we have assumed P[empty IMP] = 0; but the P[empty IMP] corresponds to the probability of being in state N (i.e., all N spaces are empty) in Fig. 3b, and thus an increase in N will not seriously affect our results.) We make the further approximation that Eq. (4) holds when σ varies as $\mu^{(k)}$, i.e., when σ is time varying. Since we have assumed overutilization, we have $(\mu^{(0)}/\sigma) < 1$, and we are justified in substituting this (or $\mu^{(k)}/\sigma$) for ρ. Thus we get the following for the approximate probability density of the length t of the time spent in the free state:

$$\rho(t) = \frac{1}{t}\sqrt{\frac{\sigma}{\mu^{(k)}}}\, e^{-(\sigma+\mu^{(k)})t} I_1\left(2t\sqrt{\sigma\mu^{(k)}}\right) \tag{5}$$

As the ratio $\mu^{(k)}/\sigma$ approaches 0, i.e., as the system becomes more overutilized, this density approaches that of the exponential distribution except out on the tail of the distribution where the probability density will be assumed negligible. To arrive at a more tractable model, we therefore approximate the free period distribution by an exponential distribution having the same mean value. The mean value of the busy period in the original system is easy to obtain, and is given by $1/\mu(1 - \rho)$. Therefore, as an approximation to the free period in the IMP, we take an exponential distribution with mean value $1/(\sigma - \mu^{(k)})$,

$$\lambda^{(k)} = \sigma - \mu^{(k)} \tag{6}$$

For the marginal case, $\sigma = \mu^{(0)}$, elementary queueing theory shows that we must take

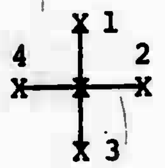
$$\lambda^{(0)} = \sigma/N \qquad \text{for } \sigma = \mu^{(0)} \tag{7}$$

where N is the size of the storage capacity of the IMP (in messages).

Our model for the blocking IMP is thus a two-state Markov process or, in the language of renewal theory, an alternating Poisson renewal process [20].

B. Derivation of the Network Model

One way to describe the dynamics of a network of such nodes is to examine the probability that any given node is blocked at some time t. For a network let us employ a two-dimensional integer lattice. In this way we can have a large system and yet minimize the complexity of its description. Consider a node with its four neighbors numbered 1 to 4:



Let

$$p^k(t) = P[k \text{ neighbors blocked at time } t] \tag{8}$$

and let

$$p(t) = P[\text{node blocked at time } t] \tag{9}$$

Then, from elementary considerations, we have (correct to within $o(\Delta t)$)

$$p(t+\Delta t) = (1-p(t))\sum_{k=0}^{4} p^k(t)\lambda^{(k)}\Delta t + p(t)\left(1-\sum_{k=0}^{4} p^k(t)\mu^{(k)}\Delta t\right)$$

where from Eq. (3)

$$\mu^{(k)} = \mu^{(0)} - (k/5)\mu^{(0)}$$

and from Eq. (6)

$$\lambda^{(k)} = \sigma - \mu^{(k)} = \sigma - \mu^{(0)} + (k/5)\mu^{(0)}$$

for $\sigma > \mu^{(0)}$. We will assume that this holds for $\sigma = \mu^{(0)}$ as well. The usefulness of the results that we will obtain will justify this approximation.

We also note that

$$\lambda^{(k)} + \mu^{(k)} = \sigma \tag{10}$$

Thus,

$$\frac{p(t + \Delta t) - p(t)}{\Delta t} = (1 - p(t)) \sum_{k=0}^{4} P^k(t)\lambda^{(k)} - p(t) \sum_{k=0}^{4} P^k(t)\mu^{(k)}$$

Letting Δt approach 0, we have

$$\begin{aligned}\frac{dp(t)}{dt} &= -p(t) \sum_{k=0}^{4} P^k(t)(\lambda^{(k)} + \mu^{(k)}) + \sum_{k=0}^{4} P^k(t)\lambda^{(k)} \\ &= -\sigma p(t) \sum_{k=0}^{4} P^k(t) + \sum_{k=0}^{4} P^k(t)(\sigma - \mu^{(0)} + (k/5)\mu^{(0)}) \\ &= -\sigma p(t) + \sigma - \mu^{(0)} + \frac{\mu^{(0)}}{5} \sum_{k=0}^{4} kP^k(t) \end{aligned} \tag{11}$$

This can be simplified by noting that

$$E[\text{number of blocked neighbors at time } t] = \sum_{k=0}^{4} kP^k(t) \tag{12}$$

where E denotes expectation. Define the indicator function

$$f_n(t) = \begin{cases} 1 & \text{if node } n \text{ is blocked at time } t \\ 0 & \text{otherwise} \end{cases}$$

Now let

$$p_n(t) = P[\text{node } n \text{ is blocked at time } t]$$

then

$$E[f_n(t)] = p_n(t) \tag{13}$$

Further, from Eq. (12) we have that

$$\sum_{k=0}^{4} kP^k(t) = E(\sum_{n \epsilon M} f_n(t)) = \sum_{n \epsilon M} E(f_n(t)) \tag{14}$$

where M is the set of neighbors for this node (which we number 1,2,3, 4). From Eqs. (13) and (14) we get

$$\sum_{k=0}^{4} kP^k(t) = p_1(t) + p_2(t) + p_3(t) + p_4(t) \tag{15}$$

Finally, from Eqs. (11) and (15) we have the result

$$\frac{dp(t)}{dt} = -\sigma p(t) + \sigma - \mu^{(0)} + \frac{\mu^{(0)}}{5}(p_1(t) + p_2(t) + p_3(t) + p_4(t)) \tag{16}$$

It is interesting that this relation can also be derived from epidemiology. We will adopt the notation from Bartlett [21].

Consider a deterministic epidemic without migration of individuals and with but two types of individuals, infected and susceptible, in which "cured" individuals are returned to the susceptible ranks. Assume that the number of individuals in an infected group who are cured at time $t + \Delta t$ is equal to the number of infectives in the group at time t multiplied by a constant, μ_0, diminished by the number of infectives found simultaneously in surrounding areas weighted in some spatial manner. Similarly, we will assume that the number of individuals in a group of susceptibles who become infected at time $t + \Delta t$ is equal to the number of susceptibles in the group at time t multiplied by a constant, λ_0, increased by a spatial weighting of the infected neighbors. Neighboring infectives will, therefore, always have a detrimental effect. They tend to increase the rate of infection and decrease the rate of cure.

Combining these assumption yields the following equation for the density of infectives at point $\underline{r}$ at time $t + \Delta t$

$$f(\underline{r},t + \Delta t) = f(\underline{r},t)[1 - \Delta t(\mu_0 - \int \mu(\underline{r} - \underline{s})f(\underline{s},t)d\underline{s}]$$
$$+ (n(\underline{r}) - f(\underline{r},t))\Delta t[\lambda_0 + \int \lambda(\underline{r} - \underline{s})f(\underline{s},t)d\underline{s}]$$

where $n(\underline{r})$ is the density of individuals of both types at $\underline{r}$, $\mu(\underline{r} - \underline{s})$ is a scalar function with a vector argument that gives the effect of infectives at $\underline{s}$ on the cure rate of infectives at $\underline{r}$, and $\lambda(\underline{r} - \underline{s})$ gives the effect of infectives at $\underline{s}$ on the infection rate of susceptibles at $\underline{r}$. Define $p(\underline{r},t)$ = P[an individual at $\underline{r}$ is infected at time t], then $p(\underline{r},t) = \dfrac{f(\underline{r},t)}{n(\underline{r})}$, and

$$p(\underline{r},t + \Delta t) = p(\underline{r},t)[1 - \Delta t(\mu_0 - \int\mu(\underline{r} - \underline{s})n(\underline{s})p(\underline{s},t)d\underline{s}]$$
$$+ (1 - p(\underline{r},t))\Delta t[\lambda_0 + \int\lambda(\underline{r} - \underline{s})n(\underline{s})p(\underline{s},t)d\underline{s}]$$

$$\frac{\partial p(\underline{r},t)}{\partial t} = -p(\underline{r},t)[\mu_0 + \lambda_0 + \int(\lambda(\underline{r}-\underline{s}) - \mu(\underline{r}-\underline{s}))n(\underline{s})p(\underline{s},t)d\underline{s}]$$

$$+ \lambda_0 + \int\lambda(\underline{r}-\underline{s})n(\underline{s})p(\underline{s},t)d\underline{s}$$

In our example system each individual (node) occupies a point on the integer lattice, and since each node is connected only to its four nearest neighbors, we have

$$\lambda(\underline{r}-\underline{s}) = \begin{cases} \lambda & \text{for } |\underline{r}-\underline{s}| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

and

$$\mu(\underline{r}-\underline{s}) = \begin{cases} \mu & \text{for } |\underline{r}-\underline{s}| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

The result is a system of differential equations which relate the probability that any node is bad at time t to the probability that other nodes are bad at time t. The equation for a non-border node at $\underline{r}$ is

$$\frac{\partial p(\underline{r},t)}{\partial t} = -p(\underline{r},t)[\mu_0 + \lambda_0 + (\lambda - \mu)(p(\underline{s}_1,t) + p(\underline{s}_2,t)$$

$$+ p(\underline{s}_3,t) + p(\underline{s}_4,t))] + \lambda_0 + \lambda(p(\underline{s}_1,t) + p(\underline{s}_2,t)$$

$$+ p(\underline{s}_3,t) + p(\underline{s}_4,t))$$

where $\underline{s}_1$, $\underline{s}_2$, and $\underline{s}_4$ are the four nearest neighbors to the node at $\underline{r}$. Values for the parameters λ_0, λ, μ_0, and μ are obtained in the following way:

$$\lambda^{(k)} = \sigma - \mu^{(0)} + \frac{k}{5}\mu^{(0)} = \lambda_0 + k\lambda$$

$$\mu^{(k)} = \mu^{(0)} - \frac{k}{5}\mu^{(0)} = \mu_0 - k\mu$$

$$\begin{cases} \lambda_0 = \sigma - \mu^{(0)} , & \lambda = \frac{\mu^{(0)}}{5} \\ \mu_0 = \mu^{(0)} , & \mu = \frac{\mu^{(0)}}{5} \end{cases}$$

Substituting these values into the differential equation yields

$$\frac{\partial p(\underline{r},t)}{\partial t} = -\sigma p(\underline{r},t) + \sigma - \mu^{(0)} + \frac{\mu^{(0)}}{5}(p(\underline{s}_1,t) + p(\underline{s}_2,t) + p(\underline{s}_3,t) + p(\underline{s}_4,t))$$

which was obtained in Eq. (16) from a strict probabilistic model.

Adjacent nodes have nearly equal probabilities of being blocked. Consider the case when all of these probabilities are exactly equal (as an approximation). Then from Eq. (16)

$$\frac{dp(t)}{dt} = -\sigma p(t) + \sigma - \mu^{(0)} + \frac{4}{5}\mu^{(0)}p(t)$$

$$= -(\sigma - \frac{4}{5}\mu^{(0)})p(t) + \sigma - \mu^{(0)}$$

which has the solution

$$p(t) = \left[p(0) - \frac{\sigma - \mu^{(0)}}{\sigma - \frac{4}{5}\mu^{(0)}}\right] e^{-(\sigma - \frac{4}{5}\mu^{(0)})t} + \frac{\sigma - \mu^{(0)}}{\sigma - \frac{4}{5}\mu^{(0)}} \qquad (17)$$

which will be assumed to hold for $\sigma \geq \mu^{(0)}$.

Now consider the alternating Poisson renewal process shown in Fig. 4. There are two states, called blocked (B) and free (F). If the

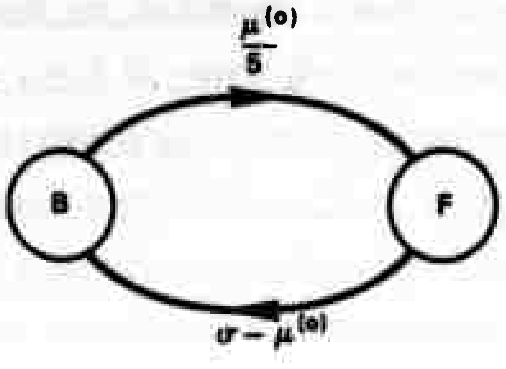


Figure 4. Network Model

system is in state B at time t, it goes to state F in the next instant Δt with probability $(\mu^{(0)}/5)\Delta t$. In similar fashion, the probability that it leaves state F and re-enters state B is $(\sigma - \mu^{(0)})\Delta t$. Therefore, the probability that it is in the blocked state at time $t + \Delta t$ is

$$p_B(t + \Delta t) = p_B(t)(1 - \frac{\mu^{(0)}}{5}\Delta t) + (1 - p_B(t))(\sigma - \mu^{(0)})\Delta t$$

$$\frac{dp_B(t)}{dt} = -p_B(t)(\sigma - \frac{4}{5}\mu^{(0)}) + (\sigma - \mu^{(0)})$$

or

$$p_B(t) = \left[p_B(0) - \frac{\sigma - \mu^{(0)}}{\sigma - \frac{4}{5}\mu^{(0)}}\right] e^{-(\sigma - \frac{4}{5}\mu^{(0)})t} + \frac{\sigma - \mu^{(0)}}{\sigma - \frac{4}{5}\mu^{(0)}} \qquad (18)$$

This is the same as Eq. (17) which was obtained for the probability that a node is blocked at time t! In a large homogeneous network, the fraction of blocked nodes may be closely approximated by the probability that any one of them is blocked. Therefore, the fraction of blocked nodes at time t in a large uniformly connected (i.e., two-dimensional lattice) network is approximately equal to the probability that the two-state Markov process shown in Fig. 4 is in the blocked state at time t. Thus we may take this two-state Markov process as a model for the network.

So far we have presented only aggregate results. To obtain the probability that any given node in the network is blocked at time t we must consider a system of equations of the form (see Eq. (16))

$$\frac{dp_i(t)}{dt} = -\sigma p_i(t) + \sigma - \mu^{(0)} + \frac{\mu^{(0)}}{5}(p_j(t) + p_k(t) + p_\ell(t) + p_m(t))$$

for each node i in the network with neighbors j, k, ℓ, and m. These equations are obviously of the form

$$\dot{P}(t) = AP(t) + C \tag{19}$$

If there are M nodes in the net, then $P(t)$ is the $M \times 1$ matrix whose i^{th} component is the probability that node i is blocked at time t. A is an $M \times M$ constant matrix and C is an $M \times 1$ constant matrix. The solution is well known:

$$P(t) = e^{At}P(0) + A^{-1}(e^{At} - I)C \tag{20}$$

For a small net this solution poses no difficulty, but for a large one the required matrix computations rapidly get out of hand. There are some special cases which are solvable, however, and we obtain the solution for one of these below.

Consider a network consisting of 1024 nodes arranged in a 32 x 32 (n x n) grid. For this system the matrix A is $n^2 \times n^2$ or 1024 x 1024 and takes the following form:

$$A = \begin{bmatrix} D & \Lambda & & & & \\ \Lambda & D & \Lambda & & \bigcirc & \\ & \Lambda & D & \Lambda & & \\ & & & \cdots & & \\ & \bigcirc & & \Lambda & D & \Lambda \\ & & & & \Lambda & D \end{bmatrix} \qquad (21)$$

$$\text{where } D = \begin{bmatrix} a & b & & & & \\ b & a & b & & \bigcirc & \\ & b & a & b & & \\ & & & \cdots & & \\ & \bigcirc & & b & a & b \\ & & & & b & a \end{bmatrix}_{n \times n} \qquad (22)$$

and

$$\Lambda = bI_n \qquad (23)$$

where

$$a = -\sigma,\ b = \frac{\mu^{(0)}}{5}, \text{ and } I_n \text{ is the } n \times n \text{ identity matrix.} \qquad (24)$$

This observation holds for a square grid with any number of nodes n on a side. (See Appendix A which gives the complete solution for P(t) with arbitrary n for this network configuration and two others.)

The network model predicts that the equilibrium fraction of blocked nodes is zero for the case $\sigma = \mu^{(0)}$. For an infinite value of N (the storage size in the IMP) this result would be obtained. However, for finite N the equilibrium fraction of blocked nodes is non-zero. To obtain an expression for this equilibrium value we must look at the different topologies of connected blocked nodes, which we call clumps. As a by-product of this analysis we will also get the clump size distribution for the case $\sigma = \mu^{(0)}$.

C. Clumping Analysis

1. Definition of a Clump

For a lattice network in which each node has exactly four neighbors (adjacent nodes) we wish to define a clump of blocked nodes. Two blocked nodes are in the same clump if they are adjacent or are linked to each other through a series of adjacent blocked nodes. A blocked node that is surrounded by four free nodes is a clump of size one.

2. Markov Chain Model for Clump Growth

Suppose $\sigma = \mu^{(0)}$ and that the expected fraction of blocked nodes is very low, say less than .1. Then the probability of the interaction of two clumps is very small, being on the order of .01, and we are justified in looking at the growth of clumps from single nodes (as an approximation). Thus we will neglect the possibility that two clumps combine. The simulation results (described later) indicate that this approximation is good for a storage size $N \geq 50$. Also, we neglect the effect of the HOSTs since, by assumption, they cannot become blocked.

Consider one free node in the midst of many free nodes. It becomes blocked in a Poisson fashion at a rate $\lambda^{(0)}$. Then we have the following situation:

```
   O
O  X  O          X = blocked node
                 O = free node
   O
```

This clump of one can become a clump of two at a rate $4\lambda^{(1)}$, again in a Poisson fashion, or die out at a rate $\mu^{(0)}$. Suppose it becomes a clump of two, then we have the following:

```
   O  O
O  X  X  O
   O  O
```

This clump of two can become a clump of three at a rate $6\lambda^{(1)}$, or become a clump of one at a rate $2\mu^{(1)}$. Suppose it goes to a clump of three, of which there are two forms:

```
I)     O  O               II)     O  O  O
    O  X  X  O                 O  X  X  X  O
       O  X  O                    O  O  O
          O
```

Form I has a growth rate of $6\lambda^{(1)} + \lambda^{(2)}$ and a death rate of $2\mu^{(1)} + \mu^{(2)}$, while form II has a growth rate of $8\lambda^{(1)}$ and a death rate $2\mu^{(1)} + \mu^{(2)}$. The death rates are obviously equal for the two different forms, but, surprisingly, the growth rates are also. Recalling that

$$\lambda^{(k)} = \sigma - \mu^{(0)} + \frac{k}{5}\mu^{(0)}$$

and $$\mu^{(k)} = \mu^{(0)} - \frac{k}{5}\mu^{(0)}$$

we have for form I (using λ_I to indicate growth rate for form I):

$$\lambda_I = 6\lambda^{(1)} + \lambda^{(2)} = 6(\sigma - \mu^{(0)} + \frac{1}{5}\mu^{(0)}) + \sigma - \mu^{(0)} + \frac{2}{5}\mu^{(0)}$$

$$= 7(\sigma - \mu^{(0)}) + \frac{8}{5}\mu^{(0)}$$

and $$\lambda_{II} = 8\lambda^{(1)} = 8(\sigma - \mu^{(0)}) + \frac{8}{5}\mu^{(0)}$$

Then, for the case $\sigma = \mu^{(0)}$, we have $\lambda_I = \lambda_{II} = 8\lambda^{(1)}$.

There are five different topologies for a clump of four blocked nodes:

```
I)        O  O
       O  X  X  O
       O  X  X  O
          O  O
```

$\lambda_I = 8\lambda^{(1)}$

$\mu_I = 4\mu^{(2)}$

```
II)       O  O  O  O
       O  X  X  X  X  O
          O  O  O  O
```

$\lambda_{II} = 10\lambda^{(1)}$

$\mu_{II} = 2\mu^{(2)} + 2\mu^{(1)}$

```
III)         O  O
          O  X  X  O
       O  X  X  O
          O  O
```

$\lambda_{III} = 6\lambda^{(1)} + 2\lambda^{(2)} = 10\lambda^{(1)}$

$\mu_{III} = 2\mu^{(2)} + 2\mu^{(1)}$

IV)
```
            O
      O  O  X  O
   O  X  X  X  O
      O  O  O
```

$$\lambda_{IV} = 8\lambda^{(1)} + \lambda^{(2)} = 10\lambda^{(1)}$$
$$\mu_{IV} = 2\mu^{(2)} + 2\mu^{(1)}$$

V)
```
      O  O  O
   O  X  X  X  O
      O  X  O
         O
```

$$\lambda_V = 6\lambda^{(1)} + 2\lambda^{(2)} = 10\lambda^{(1)}$$
$$\mu_V = 3\mu^{(1)} + \mu^{(3)} = 2\mu^{(2)} + \mu^{(1)}$$

The growth and death rates are, except for the square, form I, the same for the different forms. So, to determine the growth and death rates for a clump of four blocked nodes it is approximately sufficient to look at the straight line form, form II. For larger size clumps, we consider only this straight line form for determining the growth and death rates. Such a simplification is, of course, necessary since the number of distinct topologies prohibits exhaustive treatment. Simulation results support this approximation and show that elongated clumps are more likely to occur in systems of this kind than are square or circular-shaped clumps with their minimum circumference to area ratio. For a clump of length n we therefore have the following:

```
      O  O       O  O
   O  X  X  ...  X  X  O
      O  O       O  O
```

$$\left.\begin{aligned} \lambda_n &= 2(n+1)\lambda^{(1)} \\ \mu_n &= (n-2)\mu^{(2)} + 2\mu^{(1)} \end{aligned}\right\} \quad (25)$$

Thus our approximation leads us to the following birth-death process for clump size:

$$\begin{tikzcd}
\boxed{0} \arrow[r, bend left, "\lambda^{(0)}"] & \boxed{1} \arrow[l, bend left, "\mu^{(0)}"] \arrow[r, bend left, "4\lambda^{(1)}"] & \boxed{2} \arrow[l, bend left, "2\mu^{(1)}"] \arrow[r, bend left, "6\lambda^{(1)}"] & \boxed{3} \arrow[l, bend left, "2\mu^{(1)}+\mu^{(2)}"] \arrow[r, bend left, "8\lambda^{(1)}"] & \boxed{4} \arrow[l, bend left, "2(\mu^{(1)}+\mu^{(2)})"] \arrow[r, bend left, "10\lambda^{(1)}"] & {} \arrow[l, bend left]
\end{tikzcd}$$

$$\lambda_n = 2(n+1)\lambda^{(1)}$$

$$\mu_n = (\mu-2)\mu^{(2)} + 2\mu^{(1)}$$

We will simplify μ_n somewhat.

$$\begin{aligned}
\mu_n &= (n-2)\mu^{(2)} + 2\mu^{(1)} \\
&= n\mu^{(2)} - 2x\tfrac{3}{5}\mu^{(0)} + 2x\tfrac{4}{5}\mu^{(0)} \\
&= n\mu^{(2)} + \mu^{(3)} \\
&\approx (n+1)\mu^{(2)}
\end{aligned}$$

Therefore, to simplify the solution we use the approximations

$$\left.\begin{aligned}
\lambda_n &= 2(n+1)\lambda^{(1)} & n \geq 1 \\
\mu_n &= (n+1)\mu^{(2)} & n \geq 2
\end{aligned}\right\} \qquad (26)$$

Define p_n to be the equilibrium probability of n blocked nodes in the clump and by elementary queueing theory [7]

$$p_n = p_0 \prod_{i=0}^{n-1} \frac{\lambda_i}{\mu_{i+1}} \qquad n \geq 0$$

$$p_n = p_0 \frac{\lambda^{(0)}}{\mu^{(0)}} \prod_{i=1}^{n-1} \frac{2(i+1)\lambda^{(1)}}{(i+2)\mu^{(2)}} \qquad n \geq 1$$

$$= \frac{2p_0\,\lambda^{(0)}}{(n+1)\mu^{(0)}} r^{n-1} \qquad \text{where } r = \frac{2\lambda^{(1)}}{\mu^{(2)}} \qquad (27)$$

$$E[\#\text{ in system}] = \sum_{n=0}^{\infty} np_n = 2p_0 \frac{\lambda^{(0)}}{\mu^{(0)}} \sum_{n=1}^{\infty} \frac{n}{n+1} r^{n-1}$$

$$= 2p_0 \frac{\lambda^{(0)}}{\mu^{(0)}} \left(\sum_{n=1}^{\infty} r^{n-1} - \frac{1}{r^2} \sum_{n=1}^{\infty} \frac{r^{n+1}}{n+1} \right)$$

$$= 2p_0 \frac{\lambda^{(0)}}{\mu^{(0)}} \left(\frac{1}{1-r} - \frac{1}{r^2} \int_0^r \sum_{n=1}^{\infty} x^n dx \right)$$

$$= 2p_0 \frac{\lambda^{(0)}}{\mu^{(0)}} \left(\frac{1}{1-r} - \frac{1}{r^2} \int_0^r \frac{x}{1-x} dx \right)$$

$$= 2p_0 \frac{\lambda^{(0)}}{\mu^{(0)}} \left(\frac{1}{1-r} + \frac{1}{r} - \frac{1}{r^2} \log \left(\frac{1}{1-r} \right) \right) \quad (28)$$

p_0 is obtained in the usual way:

$$\sum_{n=0}^{\infty} p_n = 1 = p_0 \left(1 + \frac{\lambda^{(0)}}{\mu^{(0)}} \frac{2}{r^2} \left(- r - \log(1-r) \right) \right)$$

$$p_0 = \left[1 + \frac{\lambda^{(0)}}{\mu^{(0)}} \frac{2}{r^2} \left(- r - \log(1-r) \right) \right]^{-1} \quad (29)$$

and

$$p_n = p_0 \frac{\lambda^{(0)}}{\mu^{(0)}} \frac{2}{n+1} r^{n-1} \qquad n \geq 1 \qquad \text{where} \quad r = \frac{2\lambda^{(1)}}{\mu^{(2)}} \quad (30)$$

For the case $\sigma = \mu^{(0)}$ we thus have the equilibrium fraction of blocked nodes (Eq. (28)) and the distribution of clump size (Eq. (30)).

In Appendix B we apply the clumping analysis to the 8-neighbor case (shown in Fig. 5) to obtain the equilibrium fraction of blocked nodes for this network configuration when $\sigma = \mu^{(0)}$. The results obtained from the clumping analysis are good for the case $\sigma = \mu^{(0)}$ in both 4- and 8-neighbor configurations. But the results are very poor for $\sigma > \mu^{(0)}$, and this is probably attributed to the interaction of clumps. We treat this case next.

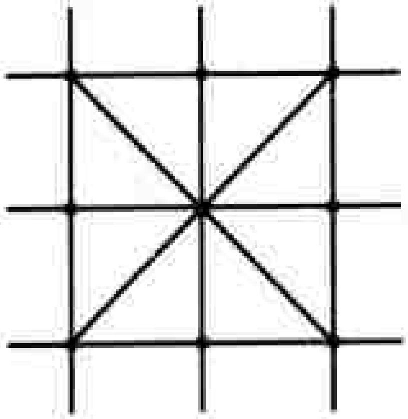


Figure 5. Eight Neighbor Lattice

3. Average Clump Size for $\sigma > \mu^{(0)}$

Although we have not arrived at a method for determining clump size distribution in the more heavily blocked cases (i.e. $\sigma > \mu^{(0)}$), we have a method which gives a crude estimate of the average clump size for these cases. It is based on the idea that any node is potentially the "origin" of a clump. The network model, Eq. (18), gives the equilibrium probability p of a blocked node for $\sigma \geq \mu^{(0)}$ while the clumping analysis, Eq. (30), gives the distribution of clump size from an isolated node (only for $\sigma = \mu^{(0)}$). To treat the case $\sigma > \mu^{(0)}$ we must combine these ideas.

We assume that clumps occur as overlaps of clumps from origin nodes which are distributed uniformly across the lattice with probability p. It would seem that a problem of conservation of blocked nodes might exist, but for estimating the average clump size this method gives good approximate results.

Let us take the left-hand extremity of a clump as its "origin." We will use a simplified clumping analysis that assumes clumps are always linear with growth or death occurring at the ends. This is generally a poor approximation, but it has the advantage that the length of a clump is then geometrically distributed and analytic results are possible. In particular, we will find the probability that a point is neither an origin nor is "covered" by a clump and call this P[empty "system"].

The relationship of this system to an infinite server queueing system (M/M/∞) will be shown. Using arguments similar to those used in [7] to get the average length of a busy period in a single server system, we will get the average length of a one-dimensional clump for the case $\sigma > \mu^{(0)}$. Finally, we will employ three different topologies for the average two-dimensional clump and/or different interpretations for the average one-dimensional clump length to get estimates of the average clump size for the two-dimensional case with $\sigma > \mu^{(0)}$.

Let us suppose that nodes are marked (blocked) with probability p, the equilibrium fraction of blocked nodes obtained from Eq. (18) by letting $t \to \infty$. Associated with each marked point is a length, geometrically distributed, which extends out to the right as in Figure 6.

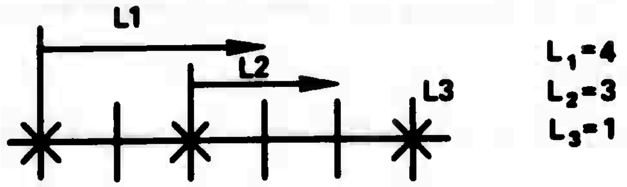


Figure 6. Clumping Model for $\sigma > \mu^{(e)}$

$$p_\ell(\ell = k) = (1 - \sigma)\sigma^{k-1} \qquad k = 1,2,\ldots$$

where $k = 1$ corresponds to a clump of size one

$$P(\ell \leq k) = 1 - P(\ell > k) = 1 - \sum_{j=k+1}^{\infty} (1 - \sigma)\sigma^{j-1} = 1 - \sigma^k$$

If a point is not covered by a line or a mark, then we say that the system (i.e. point) is "empty."

$$P[\text{empty system}] \equiv p_0 = (1 - p) \prod_{k=1}^{\infty} \left((1 - p) + p(1 - \sigma^k)\right)$$

$$= \prod_{k=0}^{\infty} (1 - p\sigma^k) \qquad (31)$$

Using

$$\log(1 - x) = -x - \frac{1}{2}x^2 - \frac{1}{4}x^3 - \frac{1}{4}x^4 - \ldots$$

we have

$$\log p_0 = \sum_{k=0}^{\infty} (-\rho\sigma^k - \frac{1}{2}(\rho\sigma^k)^2 - \frac{1}{3}(\rho\sigma^k)^3 - \ldots$$

$$= - \frac{p}{1 - \sigma} - \frac{1}{2}\frac{p^2}{1 - \sigma^2} - \frac{1}{3}\frac{p^3}{1 - \sigma^3}$$

$$p_0 = \prod_{j=1}^{\infty} \exp \frac{-p^j}{j(1 - \sigma^j)} \tag{32}$$

We make the restriction $p < 1/2$ since, analogous to the conjectured exact result for the critical probability in percolation theory (see Chapter 2, "Related Work"), the probability of an infinite clump may be non-zero for the case $p \geq 1/2$. Approximating p_0 by the first term only, we have

$$p_0 \approx e - \frac{p}{1 - \sigma} \tag{33}$$

Let us compare this model to an infinite server queueing system (M/M/∞). The average interarrival time of customers to such a system is $1/\lambda$ seconds and a customer departs after receiving an average of $1/\mu$ seconds of service. For this case we have

$$P[\text{empty system, i.e., no customers}] = e - \frac{\lambda}{\mu} \equiv P_0$$

In our nodal blocking system the "average interarrival distance" between marked points (in nodes) is

$$\frac{1}{\lambda} = p\ (1 + 2\ (1 - p) + 3(1 - p)^2 + \ldots\ )$$

$$= p \sum_{k=1}^{\infty} k(1 - p)^{k-1}$$

Let $X = 1 - p$

then $$\frac{1}{\lambda} = p \frac{d}{dX} \sum_{n=0}^{\infty} X^n = p \frac{d}{dX} \frac{1}{1 - X} = \frac{p}{(1 - X)^2} = \frac{1}{p}$$

The "average" service distance" (in nodes) is

$$\frac{1}{\mu} = \sum_{k=1}^{\infty} k(1 - \sigma)\sigma^{k-1} = \frac{1 - \sigma}{(1 - \sigma)^2} = \frac{1}{1 - \sigma}$$

and $$p_0 = e^{-\frac{\lambda}{\mu}} = e - \frac{p}{1 - \sigma} \approx p_0$$

Thus the system we are considering corresponds approximately to an infinite server queueing system.

Our system is "empty" with probability p_0 and "busy" with probability $1 - p_0$. In any line of N nodes or points $(N >> 1)$ Np_0 will, on the average, be empty and $N(1 - p_0)$ will be busy (see Fig. 6). The average length of an empty string is the average interarrival distance for our system = $1/p$ nodes. Therefore, the Np_0 empty nodes will, on the average comprise

$$\frac{Np_0}{1/p} = Np_0\, p$$

distinct empty sets or strings. Therefore, the average length of a busy string is

$$\frac{N(1 - p_0)}{Np_0\, p} = \frac{1 - p_0}{p_0\, p} \text{ nodes} \qquad (34)$$

which is analogous to the average length of a busy period in an M/M/1 queueing system [7].

We must still determine σ, the parameter in the geometric length distribution We do this by considering a line of n blocked nodes and assuming that growth or death can only occur at the ends of the string. Then we have the following:

$$\underbrace{O \quad X \quad X \quad \cdots \quad X \quad X \quad O}_{\longleftarrow\, n \,\longrightarrow} \qquad \begin{array}{l} X = \text{blocked node} \\ O = \text{free node} \end{array}$$

This implies the following birth-death process for chain length:

$$\begin{tikzcd} \boxed{0} \arrow[r, bend left, "\lambda^{(0)}"] & \boxed{1} \arrow[l, bend left, "\mu^{(0)}"] \arrow[r, bend left, "2\lambda^{(1)}"] & \boxed{2} \arrow[l, bend left, "2\mu^{(1)}"] \arrow[r, bend left, "2\lambda^{(1)}"] & {} \arrow[l, bend left, "2\mu^{(1)}"] \end{tikzcd}$$

Define $q_n = P$[chain is of length n], then

$$q_1 = \frac{\lambda^{(0)}}{\mu^{(0)}} q_0$$

$$q_n = \left(\frac{2\lambda^{(1)}}{2\mu^{(1)}}\right)^{n-1} = q_1\left(\frac{\lambda^{(1)}}{\mu^{(1)}}\right)^{n-1} \qquad n \geq 1$$

Clearly, we should take $\sigma = \dfrac{\lambda^{(1)}}{\mu^{(1)}}$ in our clump model

There are at least three possible approaches to the determination of the average clump size $\overline{C}$:

I) Assume all of the clumps are circles of radius R, and $\ell = \dfrac{1 - p_0}{p_0\, p}$ is the average length of the intersection of a random line with a circle of radius R. Kendall and Moran [22] give the average length of the intersection to be $1/2\pi R$. Then we have

$$\bar{\ell} = \frac{1 - p_0}{p_0\, p} = \frac{1}{2}\pi R \implies R = \frac{2(1 - p_0)}{\pi p_0\, p}$$

and

$$\bar{C} = \pi R^2 = \frac{1}{\pi}\left(\frac{2(1 - p_0)}{p_0\, p}\right)^2 \tag{35}$$

where $p_0 \approx e - p/(1 - \sigma)$ and p is the equilibrium fraction of blocked notes obtained from the network model.

II) Assume $\bar{l}$ is the diameter of an average clump (assumed circular), then

$$\bar{C} = \pi\left(\frac{\bar{\ell}}{2}\right)^2 \tag{36}$$

III) Assume $\bar{l}$ is the length of the side of an average clump (assumed square), then

$$\bar{C} = \bar{l}^2 \tag{37}$$

For the case which prompted this analysis all three of these methods give an average clump size within .9 of the value observed in sumulations (approximately 3.48). Method I overestimates the observed value by .76, method II underestimates it by .86, and method III underestimates it by .16.

4. <u>Maximum Clump Size</u>

A model which predicts the size of the largest clump surprisingly well was suggested to the author by Mr. Tom Leavitt of the UCLA Computer Science Department. In previous sections we assumed that "stringy" clumps are more common than round or square ones because growth in a probabilistic system occurs by shooting out projections in random directions. These random projections actually "weaken" the clump by exposing it to more free nodes. We expect the largest clumps to show a tendency to minimize their circumference with respect to their area. Therefore, in modelling the largest clumps we will use rectangular clump topologies. We assume that a clump will increase in size until the number of free nodes on the border that are becoming blocked is equal to the number of blocked nodes on the border that are becoming free. This equilibrium point corresponds to the largest clump. In order to perform the analysis we must make the following assumptions:

1) all clumps are rectangular
2) blocked nodes not on the border will remain blocked
3) every blocked node on the border has exactly three blocked neighbors
4) every free node on the border has exactly one blocked neighbor

An example of such a clump is that shown in Fig. 7.

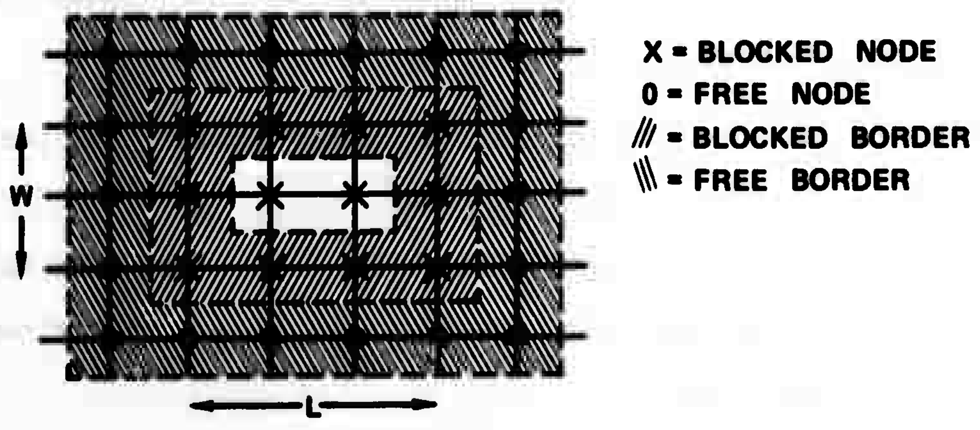


Figure 7. Maximum Clump Size Model

We see that the number of blocked nodes on the border is

$$2\ell + 2(w - 2) = 2(\ell + w - 2)$$

and the number of free nodes on the border is

$$2\ell + 2(w + 2) = 2(1 + w + 2)$$

At equilibrium we have

$$2(\ell + w - 2)\mu^{(3)}\Delta t = 2(\ell + w + 2)\lambda^{(1)}\Delta t$$

or

$$\ell + w = \frac{2(\lambda^{(1)} + \mu^{(3)})}{\mu^{(3)} - \lambda^{(1)}}$$

For a fixed border size, the number of nodes in the clump is maximized for $\ell = w$, or

$$2\ell = \frac{2(\lambda^{(1)} + \mu^{(3)})}{\mu^{(3)} - \lambda^{(1)}}$$

Therefore, the expected maximum clump size is

$$\ell^2 = \left[\frac{\mu^{(3)} + \lambda^{(1)}}{\mu^{(3)} - \lambda^{(1)}}\right]^2 \tag{38}$$

There are two reasons why this result estimates the maximum clump size and not the average clump size:

1) Clumps do not grow by adding entire borders; they add projections that weaken the clump.
2) The model assumes, incorrectly, that blocked nodes within the clump cannot become free; but they do and this further weakens the clump.

A number of models and results have been presented to characterize the behavior of a network of two-stage Markovian nodes. The efficacy of these methods will be shown in the next section in which we discuss the network simulation.

CHAPTER 4

MARKOV MODEL NETWORK SIMULATION

A. Description

Simulation of a network of 1024 nodes employing the Markovian inter-event time assumption has substantiated the analytical approximations described earlier. The two different programs which simulated this network are listed in Appendix C. These programs run on the UCLA XDS Sigma-7 computer.

The first program simulates a network arranged in a square grid 32 x 32 and simultaneously displays the net activity on a Digital Equipment Corporation 340 Precision Display CRT (Fig. 8). Each node is connected to its four nearest neighbors (a lattice) except in the case of the nodes along the border, which have only three nearest neighbors (or two nearest neighbors in the case of the four corner nodes). When a node changes state, new event times are chosen for it and for all of its nearest neighbors based on the new number of blocked neighbors. The memoryless property of the exponential distribution simplifies the calculations.

The second program simulates a randomly connected graph in which each node is given exactly four neighbors. Due to memory size limitations, this program does not have a graphical display.

B. Comparison of Observations and Predicted Behavior

1. Fraction of Nodes Blocked

Comparison of the network model (Eq. (18)) and the simulation

Reproduced from best available copy.

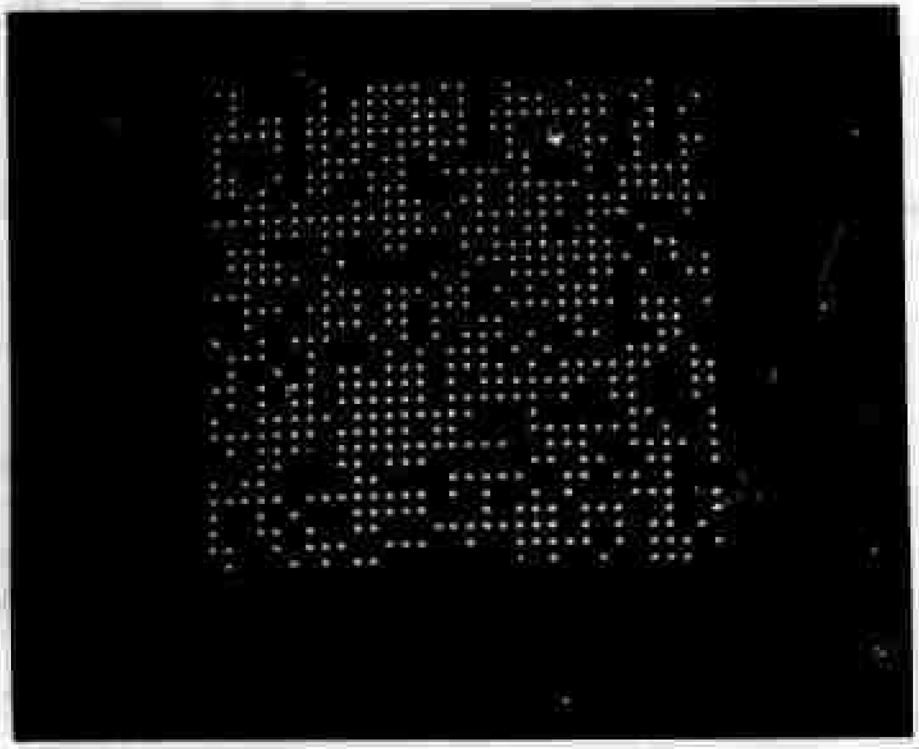


Figure 8. Network Simulation CRT Display

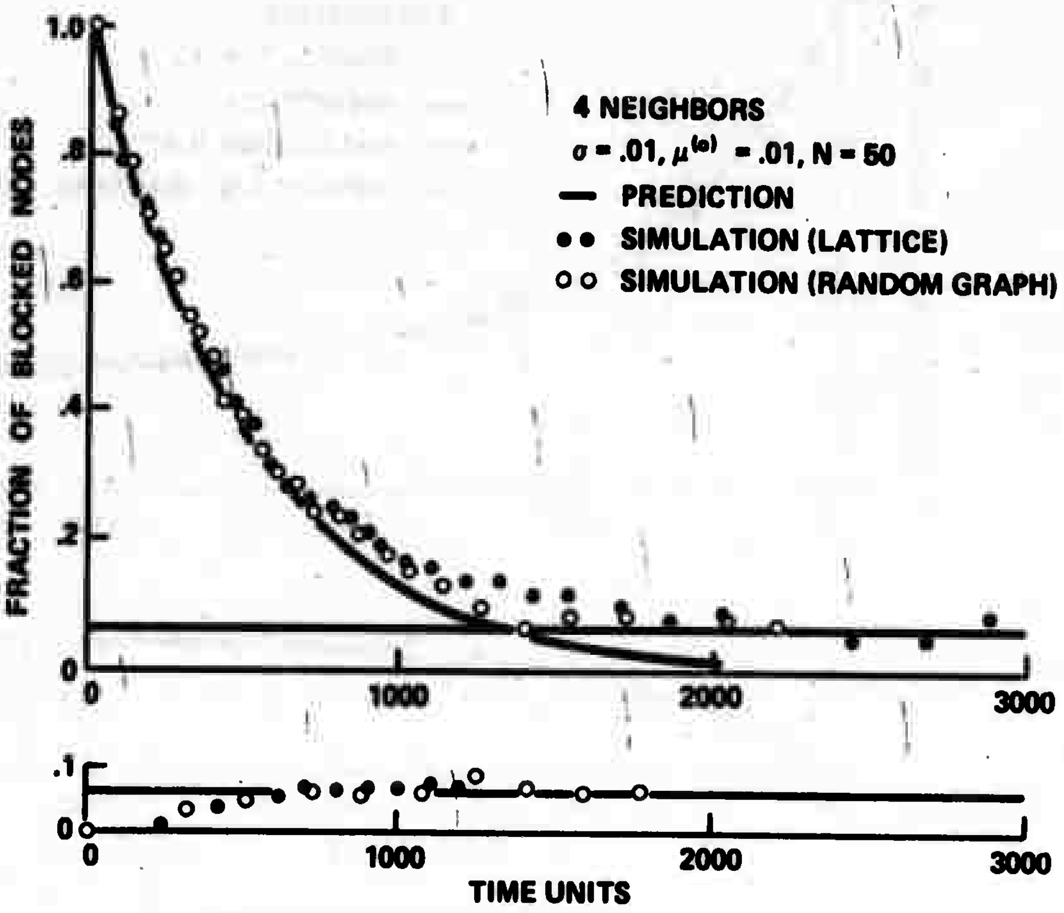


Figure 9. Fraction of Blocked Nodes I

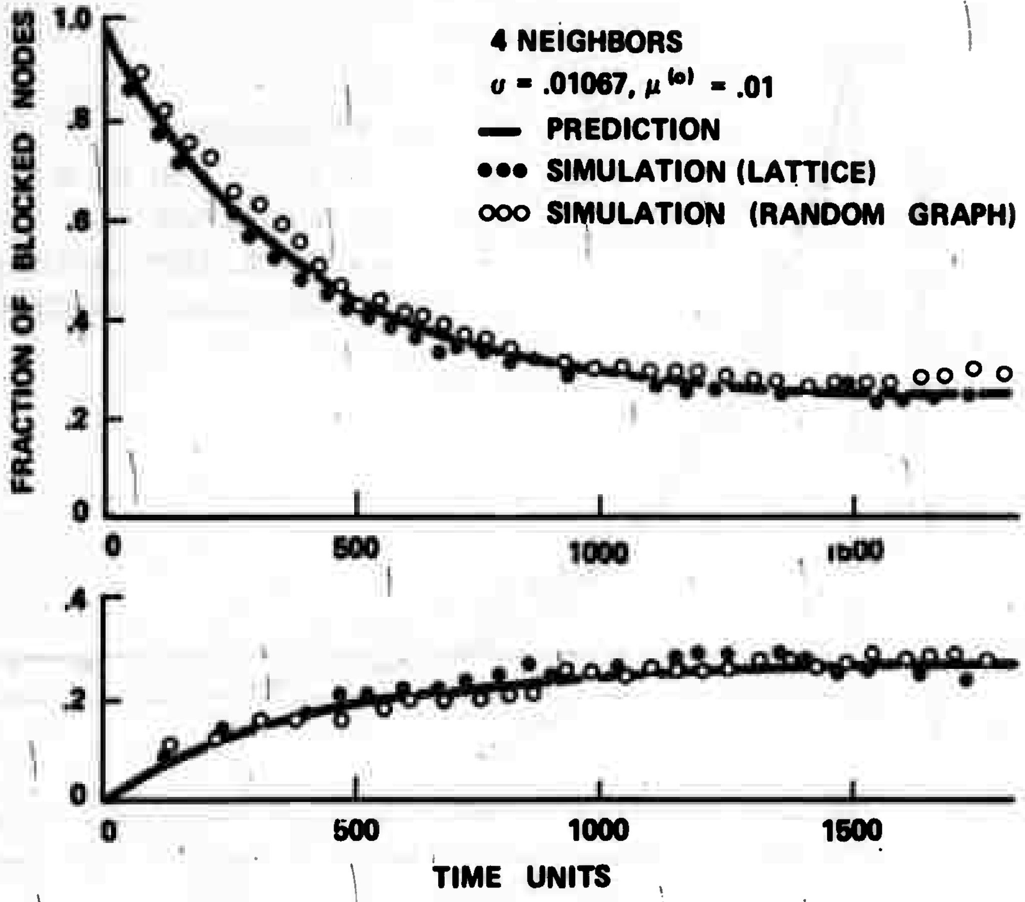


Figure 10. Fraction of Blocked Nodes II

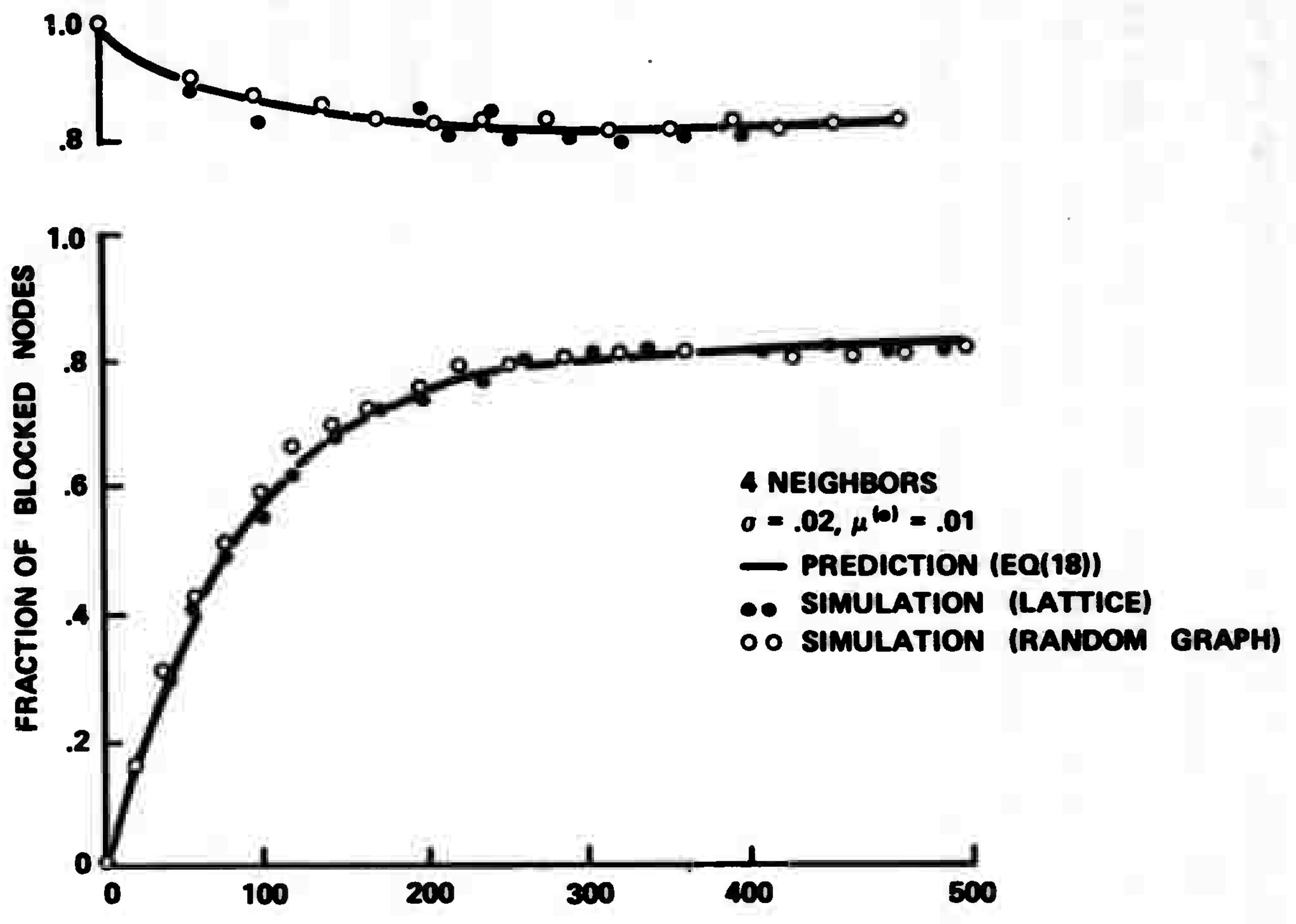


Figure 11. Fraction of Blocked Nodes III

results for the lattice and the random graph are shown in Figs. 9, 10, and 11 for three different sets of system parameters σ and $\mu^{(0)}$ each starting both from completely blocked and completely free nets. In Fig. 9 the equilibrium fraction of blocked notes is obtained from Eq. (28) of the clumping analysis.

At any point in time the network model (Eq. (18)) predicts some value f as the expected fraction of blocked notes. Assuming no correlation between nodal states and a network having 1024 nodes, β, the standard deviation of the measurement of the fraction blocked is [23]

$$\beta = \frac{\sqrt{f(1 - f)}}{32}$$

At equilibrium we have in

Figure 9:	$f = .07$	$\beta = .00796$
Figure 10:	$f = .25$	$\beta = .0135$
Figure 11:	$f = .833$	$\beta = .01165$

With a 95% confidence limit of 1.96β and a 99.7% confidence limit of 3β, we see that in Fig. 9 the assumption of independence is completely unacceptable. Recalling that the equilibrium value for this case was predicted from the clumping analysis which shows a high degree of correlation, the deviations observed in the equilibrium value in Fig. 9 are not surprising. The behavior observed in Figs. 10 and 11 is generally within the 99.7% confidence limit. Occasional excursions outside this range show the effect of clump formation and dissolution.

Figures 12, 13, and 14 give simulation results for the two-dimensional integer lattice in which each node is assumed to have eight neighbors. This was accomplished by extending the nearest neighbor defi-

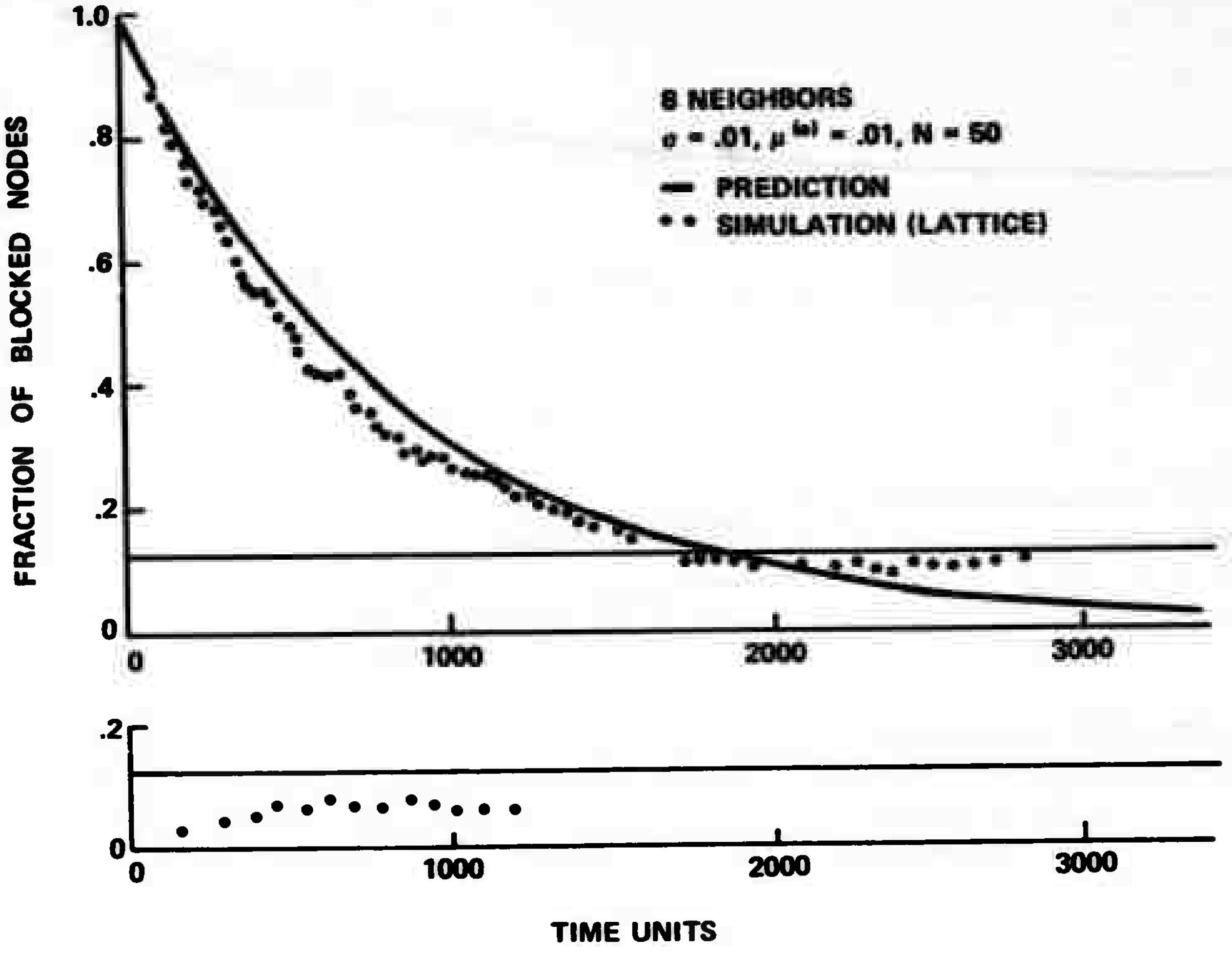


Figure 12. Fraction of Blocked Nodes IV

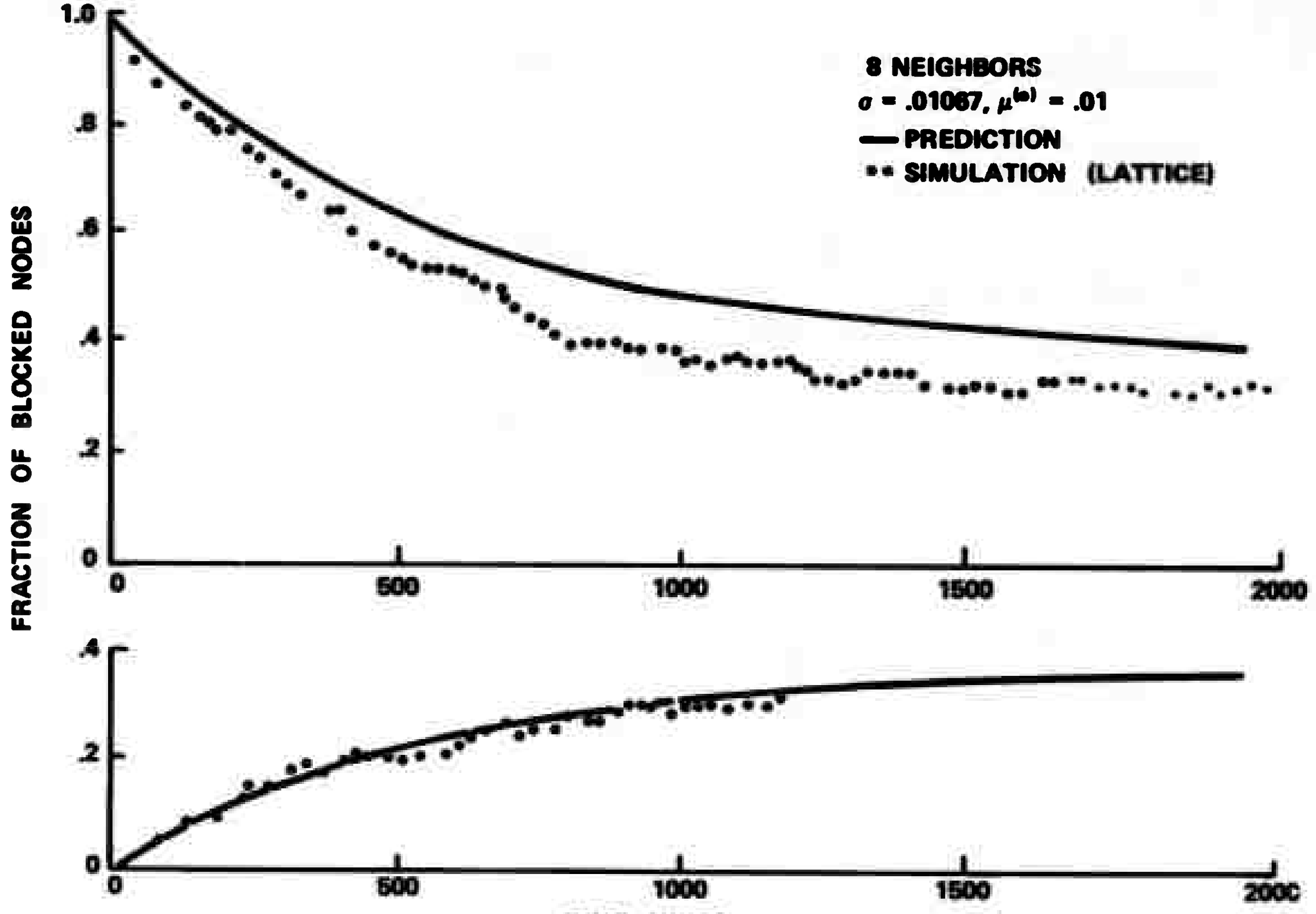


Figure 13. Fraction of Blocked Nodes V

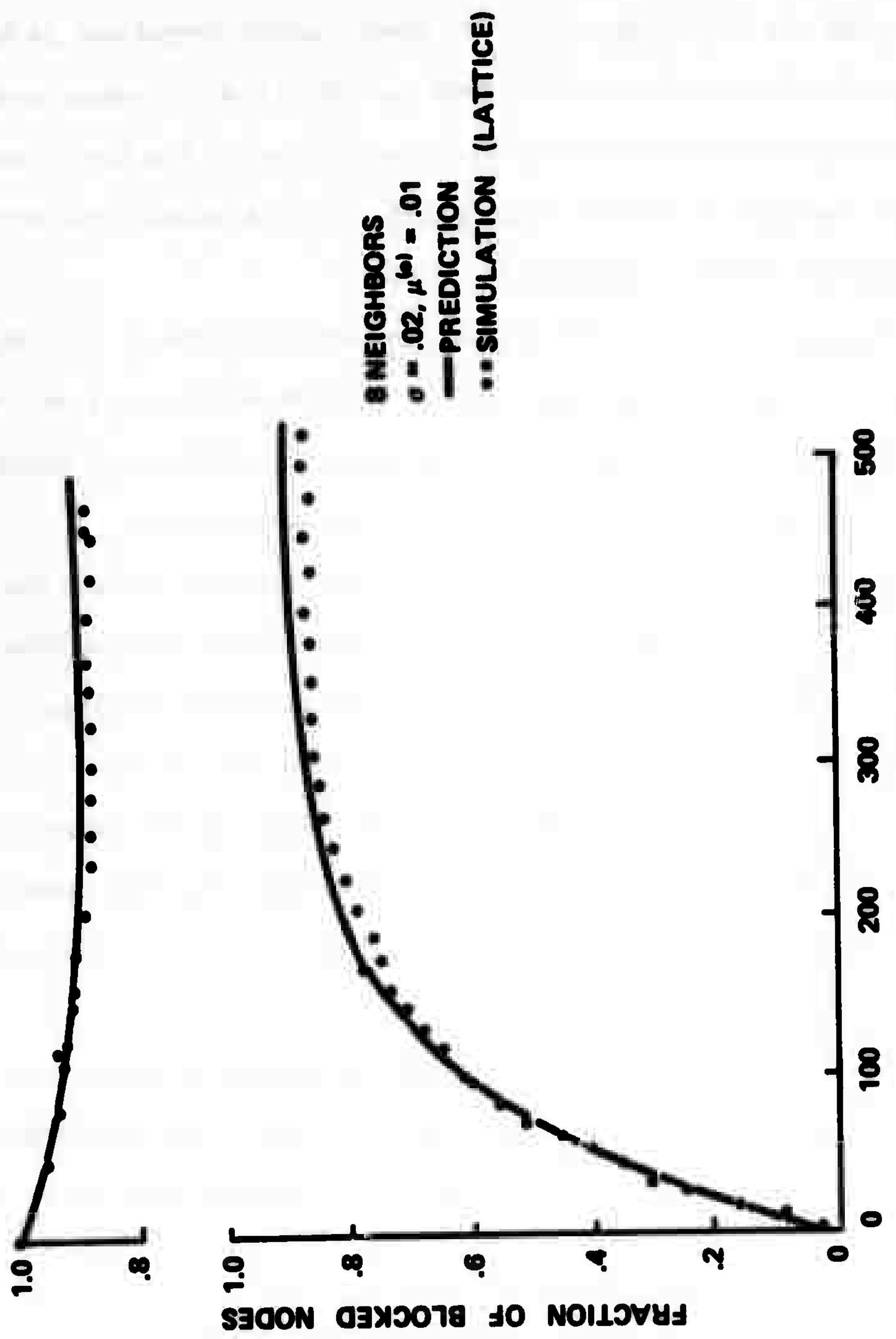


Figure 14. Fraction of Blocked Nodes VI

nition to include nodes which are diagonally adjacent. The random graph program, because of computer memory limitations, could not be modified to include the 8-neighbor case. In these figures comparison is made to the predicted behavior obtained from the network model assuming every IMP has exactly nine output lines, one of which goes to the HOST. The equilibrium fraction of blocked nodes in Fig. 12 is obtained from the clumping analysis given in Appendix B.

Figures 15, 16, and 17 compare simulation results on the lattice of degree four, when a free node with k blocked neighbors is considered k-fourths blocked, to the predicted behavior based on a non-linear "partial blocking" model. This model makes two assumptions:

1. The disturbance (i.e., blocking propagation) spreads out in a wave-like manner from blocked nodes and can be characterized as a Poisson growth process of the type studied by Morgan and Welsh [15]. In particular, we assume that the blocking starts with a single blocked node in the center of the network and that blocking is limited to what we call the "disturbed area"--those nodes which are within a distance $r(t)$ of the center node.
2. If the number of blocked nodes within the disturbed area (comprising a total of $N(t)$ nodes is $n(t)$, then the number of blocked neighbors $k(t)$ seen by an average node within the disturbed area is

$$k(t) = 4\frac{n(t)}{N(t)} + 4\left(1 - \frac{n(t)}{N(t)}\right)\frac{n(t)}{N(t)}$$

where

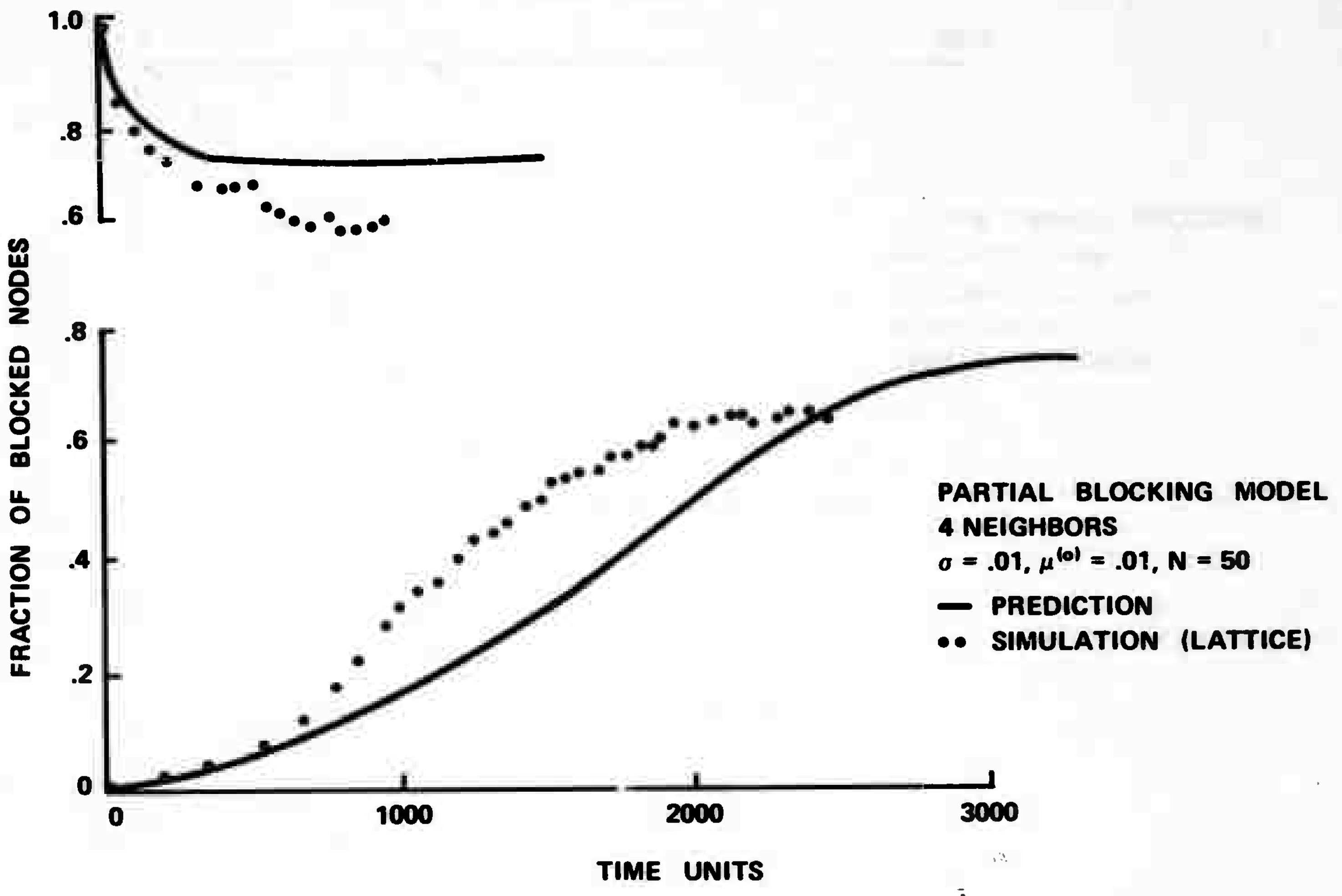


Figure 15. Fraction of Blocked Nodes VII

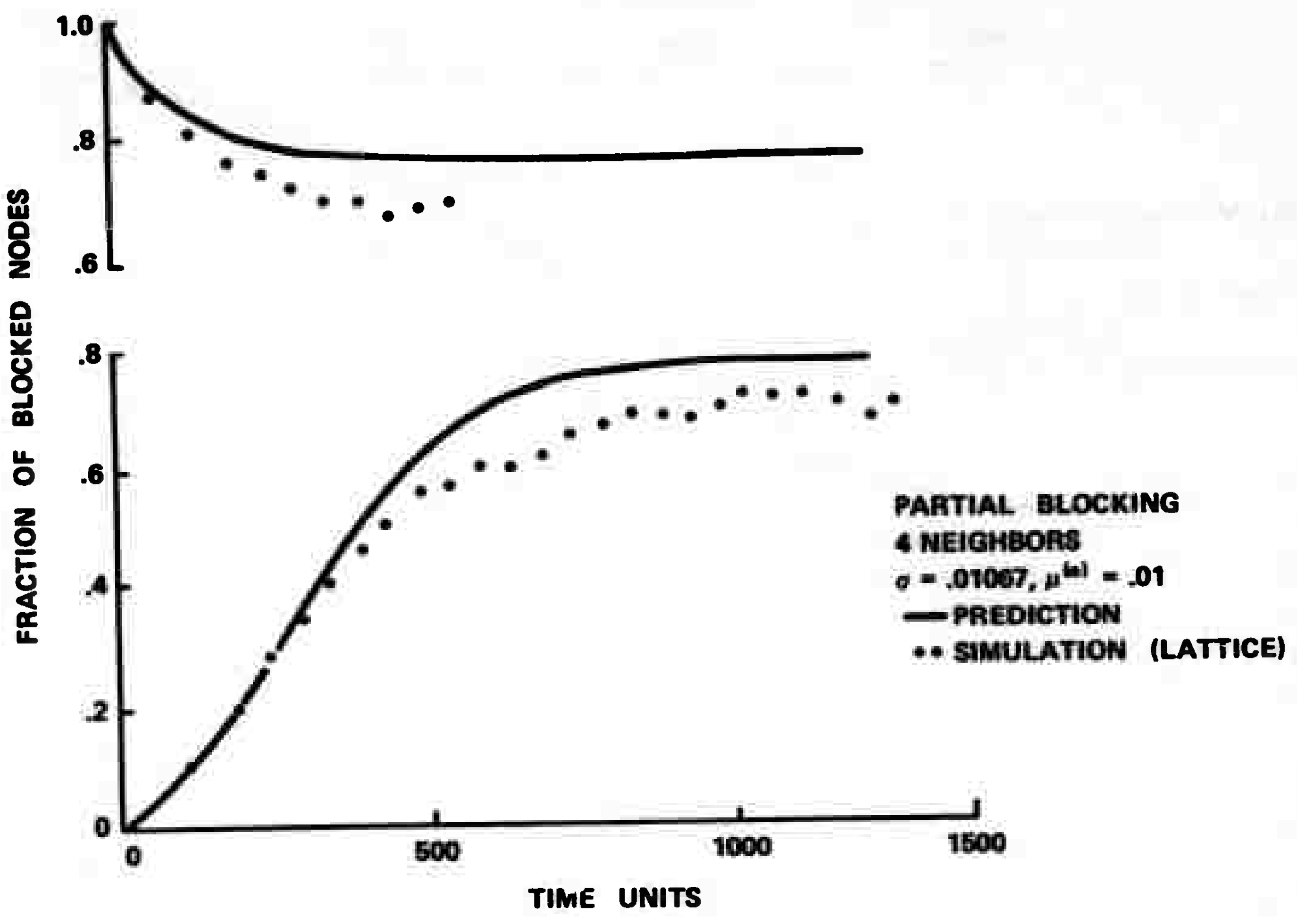


Figure 16. Fraction of Blocked Nodes VIII

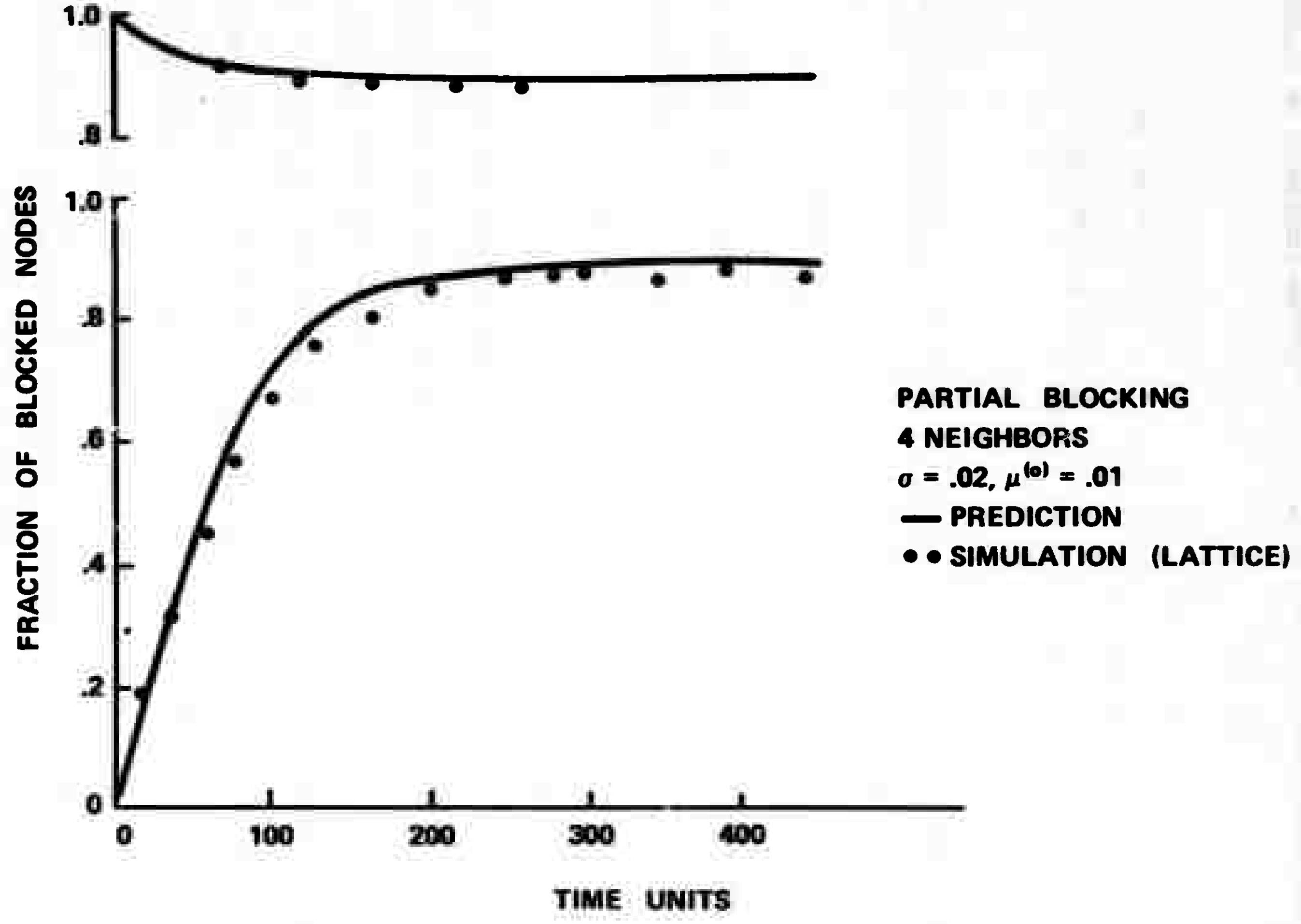


Figure 17. Fraction of Blocked Nodes IX

$$n(t + \Delta t) = n(t) - n(t)\mu^{(k(t))}\Delta t + (N(t) - n(t))\lambda^{(k(t))}\Delta t$$

$N(t)$ is found by assuming that a free node on the edge of the "disturbance wave" sees on the average 1-1/2 blocked neighbors as pictured below:

```
X  O
X  X  O            X = blocked node
X  X  O            O = free node
X  X  O
X  O
```

Then the radius of the disturbed area, $r(t)$ is given by [15] as approximately $2\lambda t$ where

$$\lambda = \lambda^{1.5} = \sigma - \mu^{(0)} + \frac{1.5}{5}\mu^{(0)}$$

These equations must be integrated step by step. The results are generally poor except in the case $\sigma = .02$, which is relatively insensitive to changes from the basic 4-neighbor network model.

2. <u>Distribution of Clump Size for $\sigma = \mu^{(0)}$</u>

Figure 18 compares the equilibrium distribution of clump size observed in the 4-neighbor lattice simulation to the prediction based on Eq. (30). Figure 19 gives the expected clump size distribution in a lattice when the blocked nodes are placed randomly on the lattice with an average fraction blocked of .07 as given by Roach [16]. We compare this to the clump size distribution observed in the random graph for the case $\sigma = \mu^{(0)}$, $N = 50$ by formally assuming that the nodes are in a lattice. The result of this assumption is a mapping that randomly disperses the clumps. The agreement is excellent, and by comparing Figs. 18 and 19 we see that the clumping in the Markov network is not at all random (i.e., uncorrelated).

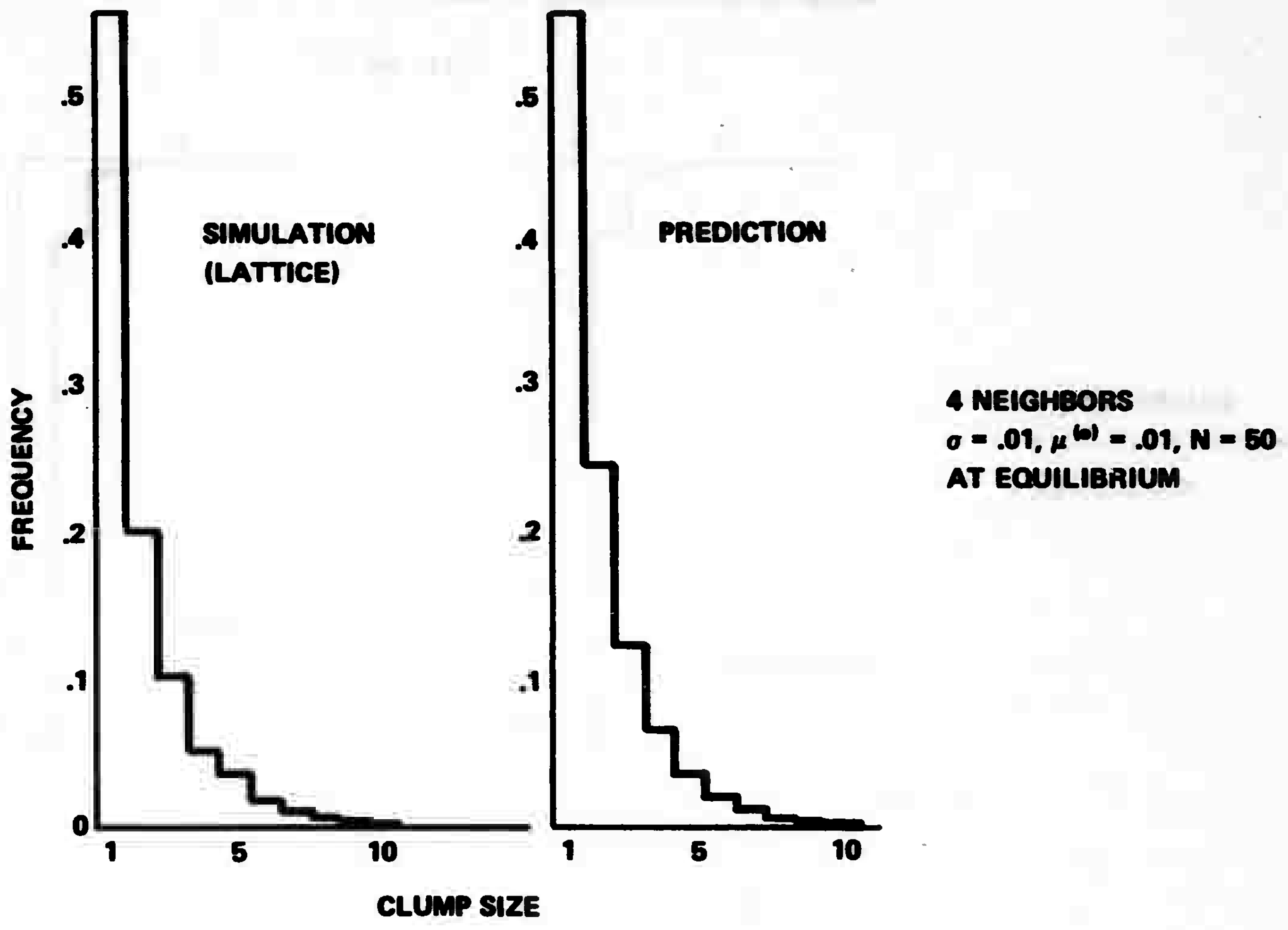


Figure 18. Clump Size Distribution

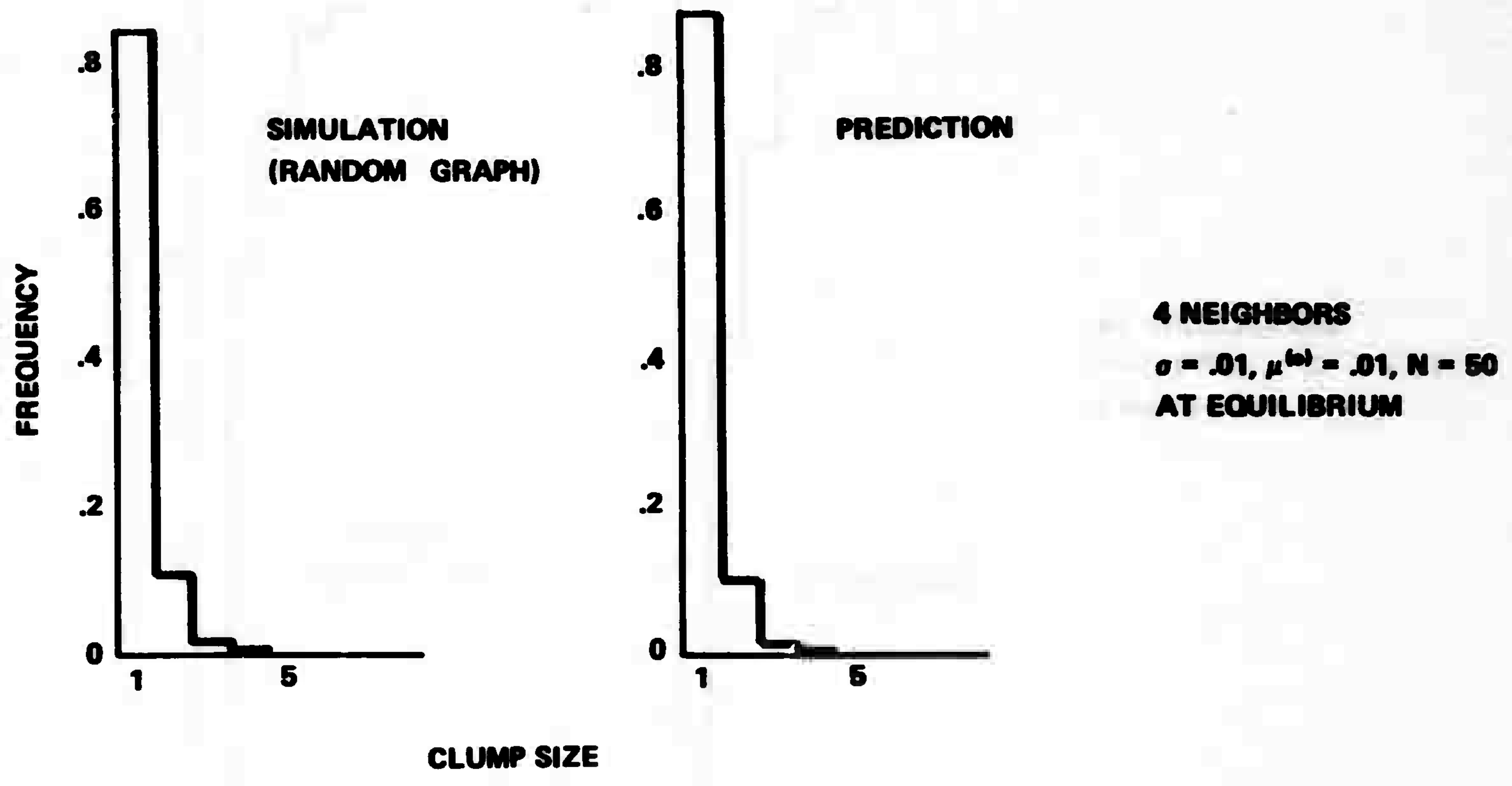


Figure 19. Random Clump Size Distribution

3. Average Clump Size for $\sigma > \mu^{(0)}$

For the case $\sigma = .01067$, $\mu^{(0)} = .01$ an average clump size of 3.48 was observed in the simulation after equilibrium was attained. The three different methods for predicting this value give estimates of 4.24, 2.62, and 3.32, respectively. Straightforward application of the clumping analysis (Eq. (28)), which is valid for $\sigma = \mu^{(0)}$, yields a value greater than 6. Hence these new methods offer some improvement.

4. Maximum Clump Size

Figures 20 and 21 show the distribution of the maximum clump size observed in the similation for two different sets of parameters after equilibrium is reached. Figure 21 shows the effect of "harmonics" of the expected maximum clump size as large clumps combined for short times. The results are remarkably good, especially considering the dispersion in the distribution in Fig. 21.

C. "Hot Spots" - Analysis and Results

In this section we analyze the effect of placing a small number of high rate of blocking (i.e. $\sigma >> \mu^{(0)}$) nodes into networks of predominantly low rate of blocking nodes ($\sigma \leq \mu^{(0)}$). We call these high rate of blocking nodes "hot spots". The simulation of a single hot spot (with $\sigma = 2\mu^{(0)}$) placed centrally in a 32 x 32 network of nodes with $\sigma = \mu^{(0)}/2$ revealed that such low rate of blocking nodes effectively prevent blocking propagation. The high rate of blocking node was the only node in the network that was ever observed to block. Hence in the analysis to follow, the low rate of blocking nodes will be assumed to have $\sigma = \mu^{(0)}$, N = 50, and we will approximate the hot spots as being permanently blocked.

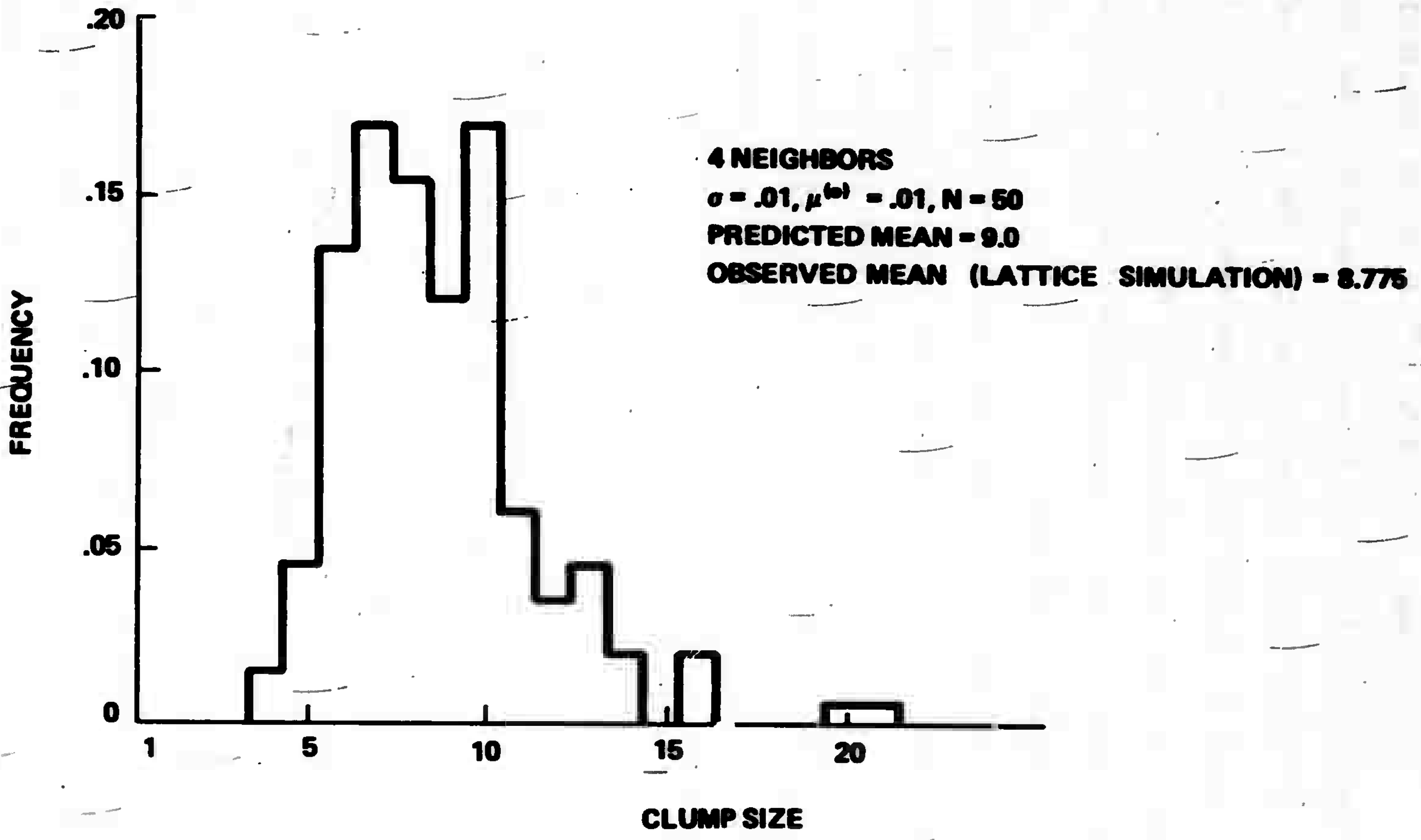


Figure 20: Maximum Clump Size Distribution I

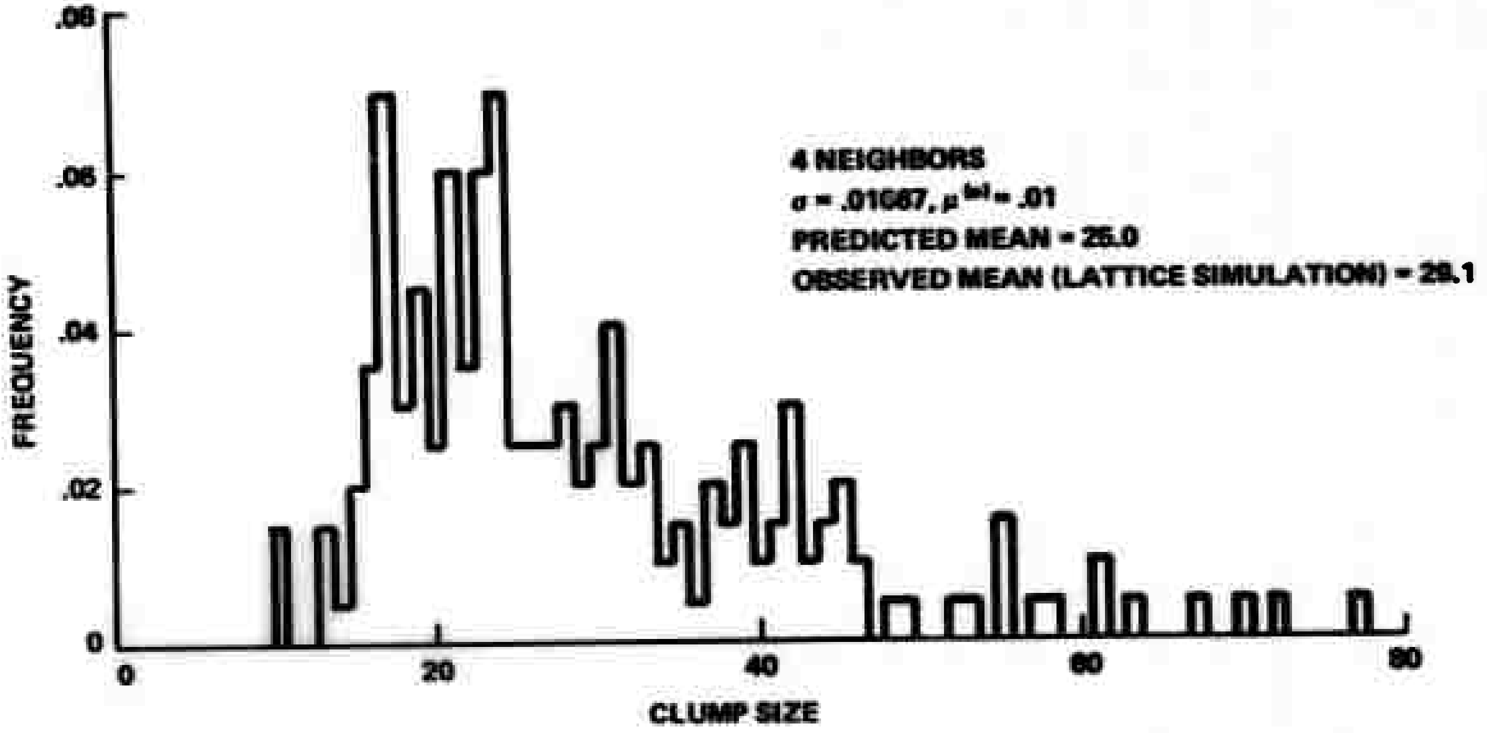


Figure 21. Maximum Clump Size Distribution II

Line of N Permanently Blocked Nodes

X   X   ...   X   X

$\xleftrightarrow{\quad N \quad}$

Suppose a line of permanently blocked nodes is put into an environment of nodes with $\sigma = \mu^{(0)}$ and $N = 50$. For a network consisting entirely of this latter kind of node, we know that the expected maximum clump size is 9 nodes. This leads us to expect a triple row of N blocked nodes, including those permanently blocked. Therefore in a net of 1024 nodes, recalling that .07 is the expected fraction of blocked nodes in the absence of permanently blocked nodes, we should have, with our blocked line,

$$\begin{aligned} E[\text{fraction blocked}] &= (3N + .07(1024 - 3N))/1024 \\ &\approx .07 \text{ for } N \text{ small} \end{aligned} \tag{39}$$

For $N = 32$, i.e., the line of permanently blocked nodes spanning the network, we should get

$$\begin{aligned} E[\text{fraction blocked}] &= (32 * 3 + .07(1024 - 32 * 3))/1024 \\ &= .157 \end{aligned} \tag{40}$$

For isolated permanently blocked nodes we must again consider the growth topologies and the Markov chain structures.

Let $\otimes$ indicate a permanently blocked node

X indicate a temporarily blocked node

$\otimes$ $\qquad\qquad$ $\lambda_1 = 4\lambda^{(1)}$

We see that from a single permanently blocked node growth occurs at a rate of $4\lambda^{(1)}$. Let us look at a clump of two and form the corresponding Markov chain:

$$X \quad \otimes \qquad\qquad \begin{aligned} \lambda_2 &= 6\lambda^{(1)} \\ \mu_2 &= \mu^{(1)} \\ \mu_3 &= 3\mu^{(1)} \end{aligned}$$

The death rate out of state 3 (i.e., a clump of 3) assumes that either of the following topologies

$$X \quad \otimes \quad X \qquad \text{or} \qquad \begin{array}{cc} & X \\ X & \otimes \end{array}$$

is much more likely than

$$X \quad X \quad \otimes$$

Already we have been forced to make approximations. The topological problems which we face in this analysis are even more difficult than those faced before in analyzing the system to obtain the average number blocked for the case $\sigma = \mu^{(0)}$. At that time we found it useful to make the approximation

$$\lambda_n = 2(n + 1)^{(1)} \qquad n \geq 1$$

$$\mu_n = (n + 1)^{(2)} \qquad n \geq 2$$

In the system with permanently blocked nodes the growth rate at different clump sizes should be the same as those above. However, the permanently blocked node cannot, by definition, become free. Assume that it is well within the clump at larger clump sizes. Then we should use the μ_n given above diminished by $\mu^{(km)}$ where km is the highest number appearing and the expression for μ_n. Hence we will assume the following growth and death rates:

$$\mu_n = 2(n + 1)\lambda^{(1)} \qquad n \geq 1$$

$$\mu_n = n\mu^{(2)} \qquad n \geq 2$$

Then

$$p_n = p_1 \prod_{i=1}^{n-1} \frac{\lambda_i}{\mu_{i+1}} \qquad n \geq 1$$

$$= p_1 \prod_{i=1}^{n-1} \frac{2(i + 1)\lambda^{(1)}}{(i + 1)\mu^{(2)}} = p_1 r^{n-1} \qquad n \geq 1$$

$$\text{where } r = \frac{2\lambda^{(1)}}{\mu^{(2)}}$$

$$\sum_{n=1}^{\infty} p_n = 1 = p_1 \sum_{n=0}^{\infty} r^n = \frac{p_1}{1 - r}$$

Therefore

$$p_1 = 1 - r$$

and

$$p_n = (1 - r)r^n \qquad n \geq 1$$

$$E[\# \text{ in system}] = \sum_{n=1}^{\infty} np_n = (1 - r)\sum_{n=1}^{\infty} nr^{n-1}$$

$$= \frac{1}{1 - r} \text{ where } r = \frac{2\lambda^{(1)}}{\mu^{(2)}}$$

For

$$\sigma = .01 = \mu^{(0)}, \; N = 50$$

$$\lambda^{(1)} = \sigma - \mu^{(1)} = \sigma - \mu^{(0)} + \frac{\mu^{(1)}}{5} = .002$$

$$\mu^{(2)} = \mu^{(0)} - \frac{2}{5}\mu^{(0)} = .006$$

Therefore

$$r = \frac{2\lambda^{(1)}}{\mu^{(2)}} = \frac{2}{3}$$

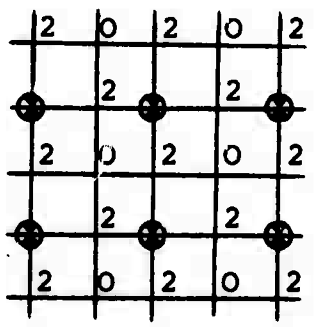
Therefore

$$E[\# \text{ in system}] = \frac{1}{1 - \frac{2}{3}} = 3$$

Thus an isolated permanently blocked node should, on the average, cause a clump of size 3 to be produced, i.e., itself plus two temporarily blocked nodes. If there are N <u>isolated</u> permanently blocked nodes and N is less than, say, 100 we should have

$$E[\text{fraction blocked}] = (3N + .07(1024 - 3N))/1024 \tag{41}$$

If N is large, i.e., greated than 100, we must iterate to a solution as in the following example. Consider a lattice of 256 permanently blocked nodes superimposed on the 1024 node network:



The number: ɛide a node indicate how many permanently blocked nodes that node has as neighbors. It is easy to see that one-third of the non-permanently blocked nodes are of the 0 type, and the other two-thirds are of the 2 type.

From the amount of time spent in the blocked state and the free state, we know that a node with X blocked neighbors is blocked with probability

$$f_x = \frac{\frac{1}{\mu}(x)}{\frac{1}{\lambda}(x) + \frac{1}{\mu}(x)} = \frac{\lambda^{(x)}}{\lambda^{(x)} + \mu^{(x)}} = \frac{\lambda^{(x)}}{\sigma}$$

Therefore, we have as a first step in the solution

$$E[\# \text{ blocked}] = 256 + 512\ f_2 + 256\ f_0$$

with

$$f_2 = \frac{\lambda^{(2)}}{\sigma} = \frac{\sigma - \mu^{(0)} + \frac{2}{5}\mu^{(0)}}{\sigma} = \frac{2}{5} \text{ and } f_0 = \frac{\sigma/50}{\sigma} = .02$$

Let k_2 = average # of blocked neighbors for a type 2 node

k_0 = average # of blocked neighbors for a type 0 node

then our iteration proceeds as follows:

$$\begin{cases} k_2 = 2 + 2 * f_0 = 2 + 2(.02) = 2.04 \approx 2 \\ k_0 = 0 + 4 * f_2 = 4(.4) = 1.6 \end{cases}$$

$$\begin{cases} f_2 = \frac{\lambda^{(k_2)}}{\sigma} = \frac{k_2}{5} = .4 \\ f_0 = \frac{\lambda^{(k_0)}}{\sigma} = \frac{k_0}{5} = .32 \end{cases}$$

$$\begin{cases} k_2 = 2 + 2 * f_0 = 2 + 2(.32) = 2.64 \\ k_0 = 0 + 4 * f_2 = 4(.4) = 1.6 \end{cases}$$

$$\begin{cases} f_2 = \frac{k_2}{5} = .528 \\ f_0 = \frac{k_0}{5} = .32 \end{cases}$$

$$\begin{cases} k_2 = 2 + 2 * f_0 = 2 + 2(.32) = 2.64 \\ k_0 = 0 + 4 * f_2 = 4(.528) = 2.112 \end{cases}$$

$$\begin{cases} f_2 = \frac{k_2}{5} = .528 \\ f_0 = \frac{k_0}{5} = .422 \end{cases}$$

$$\begin{cases} k_2 = 2 + 2 * f_0 = 2 + 2(.422) = 2.84 \\ k_0 = 0 + 4 * f_2 = 4(.528) = 2.112 \end{cases}$$

$$\begin{cases} f_2 = \frac{k_2}{5} = .569 \\ f_0 = \frac{k_0}{5} = .422 \end{cases}$$

We will end the iteration at this point and get as an approximate solution

$$E[\text{fraction blocked}] = (256 + 512\ f_2 + 256\ f_0)/1024$$

$$= .639$$

In the limit the E[fraction blocked] = .66176. (42)

The last case which we will consider is that of an R X R clump of permanently blocked nodes, with $R \geq 2$. Modelling the border of this clump as a line of permanently blocked nodes formed into a square, we should expect the clump to increase to (R + 1) X (R + 1).

Therefore,

$$E[\text{fraction blocked}] = ((R + 1)^2 + .07(1024 - (R + 1)^2))/1024 \quad (43)$$

Table 1 lists the results observed in the simulation of hot spots on the 32 x 32 grid for the following cases:

1. Two hot spots side by side
2. Two hot spots separated by one low rate of blocking node
3. Three hot spots in a connected straight line
4. 32 hot spots in a line (one whole row of the network)
5. A lattice of 64 hot spots spread evenly over the 32 x 32 grid
6. A lattice of 256 hot spots spread evenly over the grid
7. Four hot spots in a 2 x 2 clump
8. Nine hot spots in a 3 x 3 clump
9. 25 hot spots in a 5 x 5 clump

Case	% Blocked High	Time of High	% Blocked Average	Total Observation Time	% Blocked (Prediction)	Pertinent Equation
1	10.0	1327	7.5	1636	7	39
2	8.4	953	7	1331	7	39
3	8.6	1901	7	1985	7	39
4	16.8	2412	13.5	3581	15.7	40
5	25.4	838	24	1265	24.4	41
6	64.3	632	63	758	66.2	42
7	9.6	1897	7	2060	7	43
8	10.7*	2237	8.4*	2380	7	43
9	10.7	1731	9.2	2098	10.2	43

HOT SPOTS RESULTS

TABLE 1

*The high value and the overall greater average were due to the formation of a large clump that was not connected to the 3 x 3 clump.

These models have clearly proven their applicability. This completes our analysis of hot spots.

So far we have permitted ourselves the strong assumption of two-state Markovian nodes. In the next section we treat the application of these results to a simulated computer-communication network of 64 nodes which has many real world properties.

CHAPTER 5

SIMULATION OF A NETWORK WITH MESSAGE TRANSFER

A. Description

A program which simulates a store-and-forward communication network of 64 nodes was run on the UCLA XDS Sigma-7 computer (see Appendix C for a listing of this program). In this network messages are sent from origin to destination nodes under nearly fixed routing strategies. The essential characteristics of this simulation network are the following:

1. Nodes are arranged in an 8 x 8 grid and are numbered consecutively from 1 to 64 by rows. Any node i is connected to nodes $i \pm 1$, $i \pm 8$ modulo 64. The result is a "twisted torus," which shows complete symmetry for each node. (A torus network prevents the center of the net from becoming a bottleneck, and a "twisted torus" is conveniently programmed.)

2. Message lengths are exponentially distributed with an average of $5/\mu^{(0)}$ units.

3. Every node has storage for exactly N messages $(1 \leq N \leq 50)$.

4. The arrival rate of requests for inputs to the IMP from the HOST is $(\sigma - 4\mu^{(0)}/5)$.

5. When a blocked node becomes free, each of its neighbors who has a message for it makes a request to send that message to it at a rate of σ RETRY (or just σ RE).

6. Routing is fixed. The routing algorithm, after being queried by a node, relays to that node the "best" next node and the "second best"

Preceding page blank

next node for that message based on its final destination. However, every queue within a node for an output line from that node is limited in length to $N/4 + 1$. This avoids the "deadly embrace" that could result if two adjacent nodes should fill up with messages for the other and thus both become permanently blocked.

7. Messages are sent to and from the HOST on lines equal in capacity to an IMP-IMP line.

8. Message destinations are chosen within a node from a uniform distribution on the remaining 63 nodes.

With these assumptions the network was simulated with $\mu^{(0)} = .01$, $N = 50$, and various values of σ and σ RE.

B. Observations

The surprising result of these simulations was that eventually, the network blocked completely in every case observed for $\sigma \geq \mu^{(0)}$. The network in the case $\sigma = \mu^{(0)}$, did show a degree of stability, however, requiring an extremely long time to block completely. After the network had blocked completely, an inspection of the contents of the nodes showed that each was filled with messages destined for the other IMPs, i.e., they contained no HOST messages. An explanation and model for this behavior and the complete blocking of the network is given in the next section.

C. Derivation of the Modified Network Model

The basic reason that the IMPs become completely filled with messages for the other IMPs can be stated very simply. In a non-blocking network an equilibrium exists between the input-output rates (and the average storage required) for both HOST and non-HOST traffic. Blocking causes a decrease in the output rate of non-HOST messages while the input of such messages remains constant. On the other hand, blocking has no effect on either the input or the output rate of HOST traffic. The loss of equilibrium between the input and output rates for non-HOST traffic causes a gradual increase in the storage required for such traffic. Eventually, the storage is completely taken over by non-HOST traffic, and thus the rate at which the network delivers messages to destinations (HOSTS) goes to zero.

We now present a mathematical model for this phenomenon. Consider once more the simplified network model shown in Fig. 4.

The rate at which the system becomes free is $\mu^{(0)}/5$, which is equal to the average rate of message transmission into the HOST. Similarly, the rate at which the system becomes blocked is $\sigma - \mu^{(0)}$ which is the excess of the arrival rate over the total service rate. This model assumes that there is always a message in the IMP that is destined for the HOST. In real networks such may not be the case.

Let $P(t) = P$[there is a message in the IMP destined for the HOST at time t]. Then the average rate of transmission into the HOST is $p(t)\mu^{(0)}/5$ and a better network model would be that shown in Fig. 22.

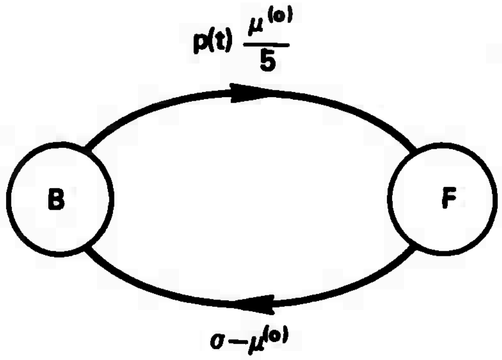


Figure 22. Modified Network Model

This model yields the following system equation:

$$\frac{dp_B(t)}{dt} = p_B(t)\left(\sigma - \mu^{(0)} + \frac{\mu}{5}^{(0)} p(t)\right) + \sigma - \mu^{(0)}$$

where $p_B(t) = P$[system is in state B (blocked)].

Before solving this equation we must derive an expression for $p(t)$, which we do by employing the Ehrenfest model of diffusion [24]. We will make the optimistic assumption that the IMP is completely filled with messages (optimistic because it increases the change of finding a message in the IMP that is destined for the HOST) and neglect the fact that this means it is blocked. We will use the modified network model (Fig. 22) to get the fraction of blocked nodes given $p(t)$, and $p(t)$ will be determined at the same time by means of the blocking history. We will then solve this system of equations.

Suppose that we have two barrels labeled HOST (H) and Store-and-Forward (SF). Distributed between these two barrels are N marbles (messages). At random times* one or the other of these barrels is chosen according to some probability law, and a marble is taken from that barrel (if it has a marble). With some probability the marble is put into the SF barrel and with the complementary probability it is put in the H barrel.

The state of the system is the number of marbles in the SF barrel at time t, or equivalently, the number of storage cells required for store-and-forward traffic. In particular, we want to know

$$p_N(t) = P[\text{barrel SF contains all N marbles}]$$

*The interval between these times is presumed to have an average value equal to the average time required for a transmission plus an arrival given the condition of the network, i.e., the fraction of blocked nodes.

which corresponds to the case of a node with no traffic deliverable to its HOST. Then it follows that

$$p(t) = 1 - p_N(t)$$

Choosing barrel H and withdrawing a marble from it represents the transmission of a message to the HOST. If there is a message to be transmitted, the transmission rate is $\mu^{(0)}/5$ (more generally it is $M_1\mu^{(0)}/M$, if there are M output lines of which M_1 go to the HOST). Choosing barrel SF and taking a marble from it represents the transmission of a store-and-forward message. If a fraction $f(t)$ of the nodes are blocked at time t, then the average output rate for store-and-forward traffic is $4/5\mu^{(0)}(1 - f(t))$ assuming that there are at least four store-and-forward messages in the IMP and all of the output lines to other IMPs are being utilized. The total output rate from the IMP is thus

$$\frac{\mu^{(0)}}{5} + \frac{4}{5}\mu^{(0)}(1 - f(t)) = \mu^{(0)} - \frac{4}{5}\mu^{(0)}f(t)$$

The probability of choosing barrel H given that there is a marble in H and at least four marbles in SF is thus

$$\frac{\mu^{(0)}/5}{\mu^{(0)} - \frac{4}{5}\mu^{(0)}f(t)}$$

and the probability of choosing barrel SF under the same conditions is

$$\frac{\frac{4}{5}\mu^{(0)}(1 - f(t))}{\mu^{(0)} - \frac{4}{5}\mu^{(0)}f(t)}$$

For the case of j SF messages in the IMP with $j < 4$, the output rate for SF traffic is $j/4$ times the output rate for $j = 4$. The total output rate and the probability of choosing a barrel must then be adjusted.

Assume that the average path length in the network is L. Then, on the average, a message visits $L + 1$ IMPs in making its way through the network. If the time spent in any segment of the path is approximately the same for all segments, then the probability of a message being in any particular segment of its path is $1/(L + 1)$. In particular, the probability that a message is in its final path segment is $1/(L + 1)$.

Placing a marble into barrel H represents the arrival of an IMP of a message that is destined for the HOST, which occurs with probability $1/(L + 1)$. Similarly, placing a marble into barrel SF represents the arrival of a store-and-forward type message, and this event occurs with probability $L/(L + 1)$.

Let us define

$P_{HA}(t)$ = P[HOST type message arrival] = P[placing a marble in barrel H]

$P_{SA}(t)$ = P[store-and-forward type message arrival]
= P[placing a marble in barrel SF]

$P_{HT}(t)$ = P[message transmission to HOST] = P[taking a marble from barrel H]

$P_{ST}(t)$ = P[message transmission to another IMP]
= P[taking a marble from barrel SF]

E_j = event that there are j marbles in barrel SF

$a_{ij}(t)$ = P[going from E_i to E_j in one step, i.e., one message transmission plus one message arrival]

If there are no store-and-forward messages in the IMP, then the probability of a transmission to the HOST is one, and if the IMP is completely filled with store-and-forward messages, the probability of a store-and-forward transmission is one. Analogously, we are not allowed to choose an empty barrel from which to withdraw a marble. As a result we get the following:

$$a_{01}(t) = p_{SA}(t)$$

$$a_{00}(t) = p_{HA}(t)$$

$$a_{jj}(t) = p_{HA}(t)p_{HT}(t) + p_{SA}(t)p_{ST}(t)$$

$$a_{jj-1}(t) = p_{ST}(t)p_{HA}(t)$$

$$a_{jj+1}(t) - p_{HT}(t)p_{SA}(t)$$

$$a_{NN-1}(t) = p_{HA}(t)$$

$$a_{NN}(t) = p_{SA}(t)$$

where, for simplicity, we have not listed all of the cases a_{ij} for i or $j < 4$.

Let

$$A(t) = [a_{ij}(t)]$$

$$p_j(t) = P[E_j \text{ at time } t)$$

and

$$P(t) = [p_0(t), p_1(t), p_2(t) \ldots p_N(t)]$$

then

$$P(t + \Delta t) = P(t)A(t)$$

We have assumed that the IMP is completely filled with messages; therefore, we must have a message departure before we can allow a message arrival. Thus, given the fraction of blocked nodes in the network

$f(t)$, and a message arrival rate to the IMP of σ message/sec., we have that the average time required for one step (1 departure + 1 arrival) is

$$\Delta t = \frac{1}{\mu^{(0)} - \frac{4}{5}\mu^{(0)} f(t)} + \frac{1}{\sigma} \tag{44}$$

The other equations comprising the system of equations that must be solved to get $p(t)$ are the following:

$$\frac{df(t)}{dt} = - f(t)\left(\sigma - \mu^{(0)} + \frac{\mu}{5}^{(0)} p(t)\right) + \sigma - \mu^{(0)} \tag{45}$$

$$P(t + \Delta t) = P(t)A(t)$$

$$p(t + \Delta t) = 1 - p_N(t + \Delta t) \tag{46}$$

To actually calculate the solution to this system of equations we must be given the initial values $p_i(0)$ and $f(0)$. Equation (45) is integrated step by step using a value of $p(t)$ that remains constant for a length of time Δt given by Eq. (44) whereupon it is recalculated using Eq. (46) with the new values of $a_{ij}(t)$.

The solution of this set of equations shows that the fraction of blocked nodes changes very slowly and the final value is higher than that preducted by the unmodified network model (Fig. 4). If state N is made an absorbing state, i.e., once the IMP becomes filled with store-and-forward messages it remains in that state, the model predicts that the network blocks completely for the case $\sigma > \mu^{(0)}$ with probability one.

D. Comparison to Simulation

The modified network model predicts that the case $\sigma = \mu^{(0)}$ should be stable, i.e. should not block completely. This is a weakness in the model. By the clumping analysis we know that the equilibrium fraction of blocked nodes in a network with these parameters and $N = 50$ should be about .07. Any amount of blocking will result in a loss of equilibrium between the input and output rates of store-and-forward traffic and thus we expect complete network blocking to be the final result.

In one simulation run with $\sigma = \mu^{(0)}$ and $N = 50$ the network stayed in the range 3.1% to 12.5% blocked for 98,000 time units. This may be compared with a time of 2,000 units which was the time required to reach equilibrium in the unmodified network model (Eq. (18)) for this set of parameters. This message transfer simulation required a net time of 250,000 time units to block completely, the net time being the amount of time from first observed blocking until the entire net is blocked. A subsequent run required 180,000 time units to block completely. Both of these runs used a value of 1,000 for σ_{RE}. The effect of this value was almost to insure that when an IMP becomes free its empty spot gets filled with a message from another IMP, which may be a HOST message. This tends to free the net. When σ_{RE} was decreased to a value of .002, a rate comparable to that at which messages are arriving from the HOST, the net time to total blocking dropped to 91,000 time units. A further decrease of σ_{RE} to 10^{-6} caused this net time to drop to 66,000 time units.

A simulation run with $\sigma = .01067$, $\mu^{(0)} = .01$, and $\sigma_{RE} = 1,000$ (Fig. 23) again showed some stability. The net time to complete blocking was observed to be 118,000 time units. Reduction of σ_{RE} to .002 (Fig. 24) and then to 10^{-6} (Fig. 25) caused the net time to drop to

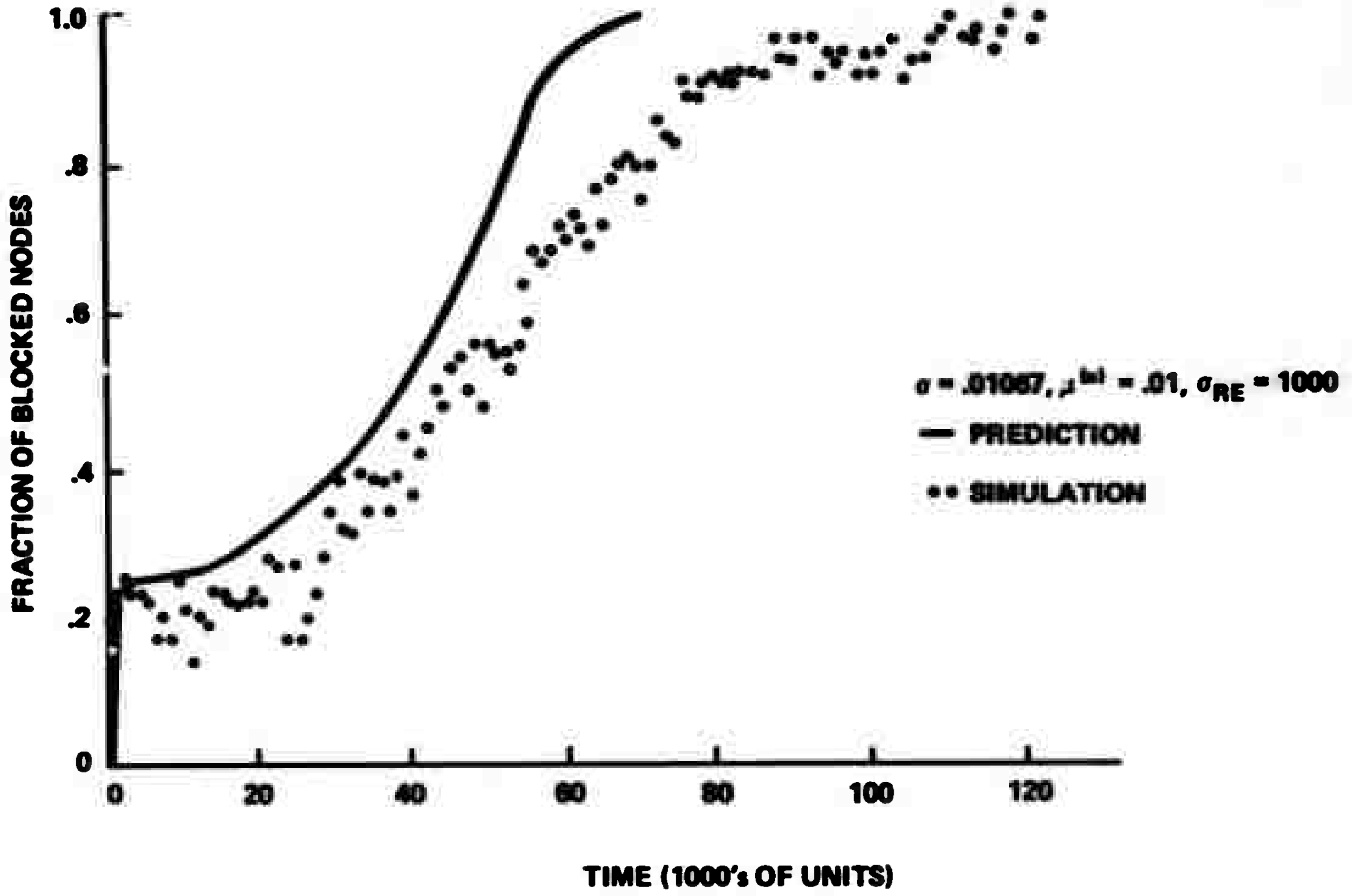


Figure 23. Store-and-Forward Network I

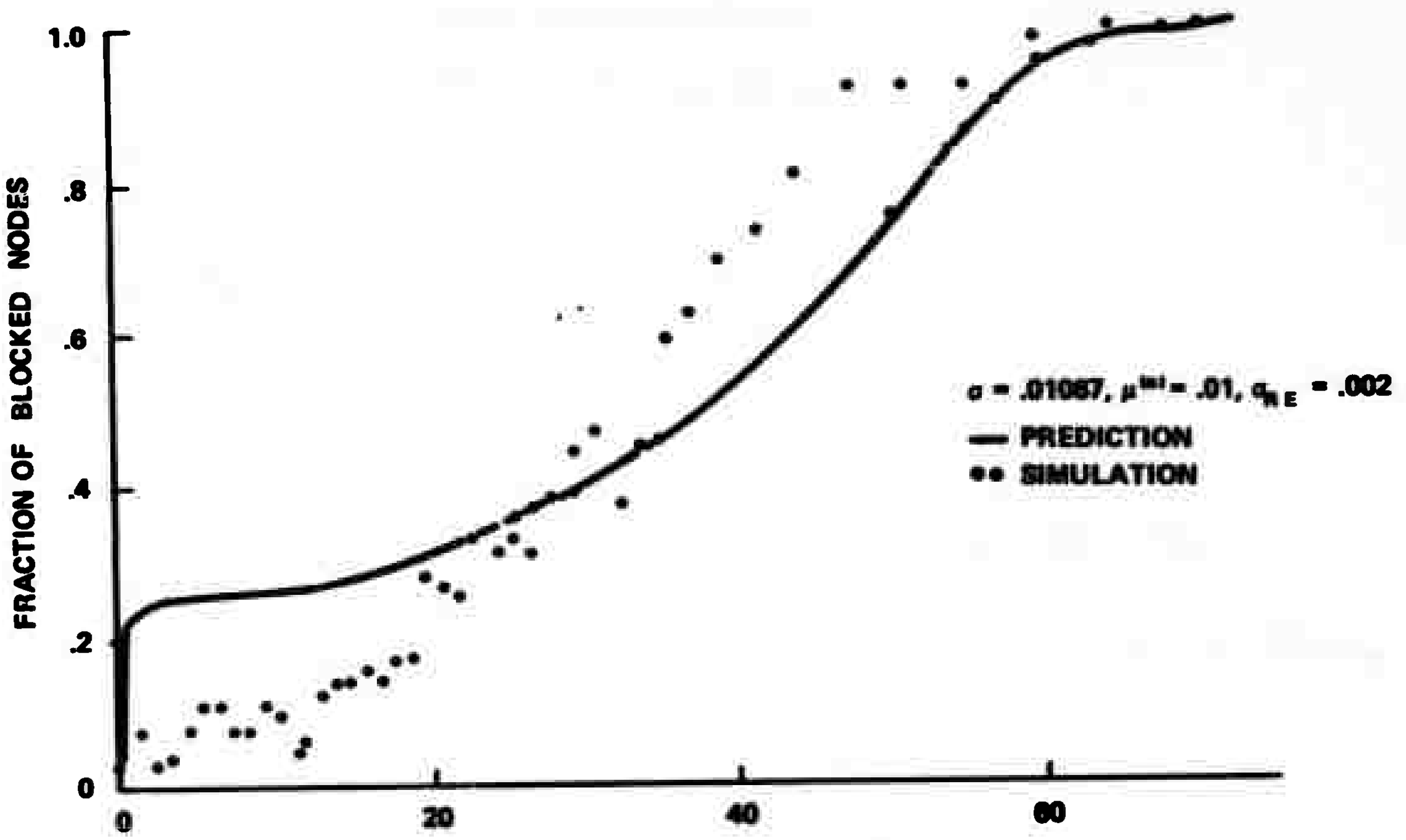


Figure 24. Store-and-Forward Network II

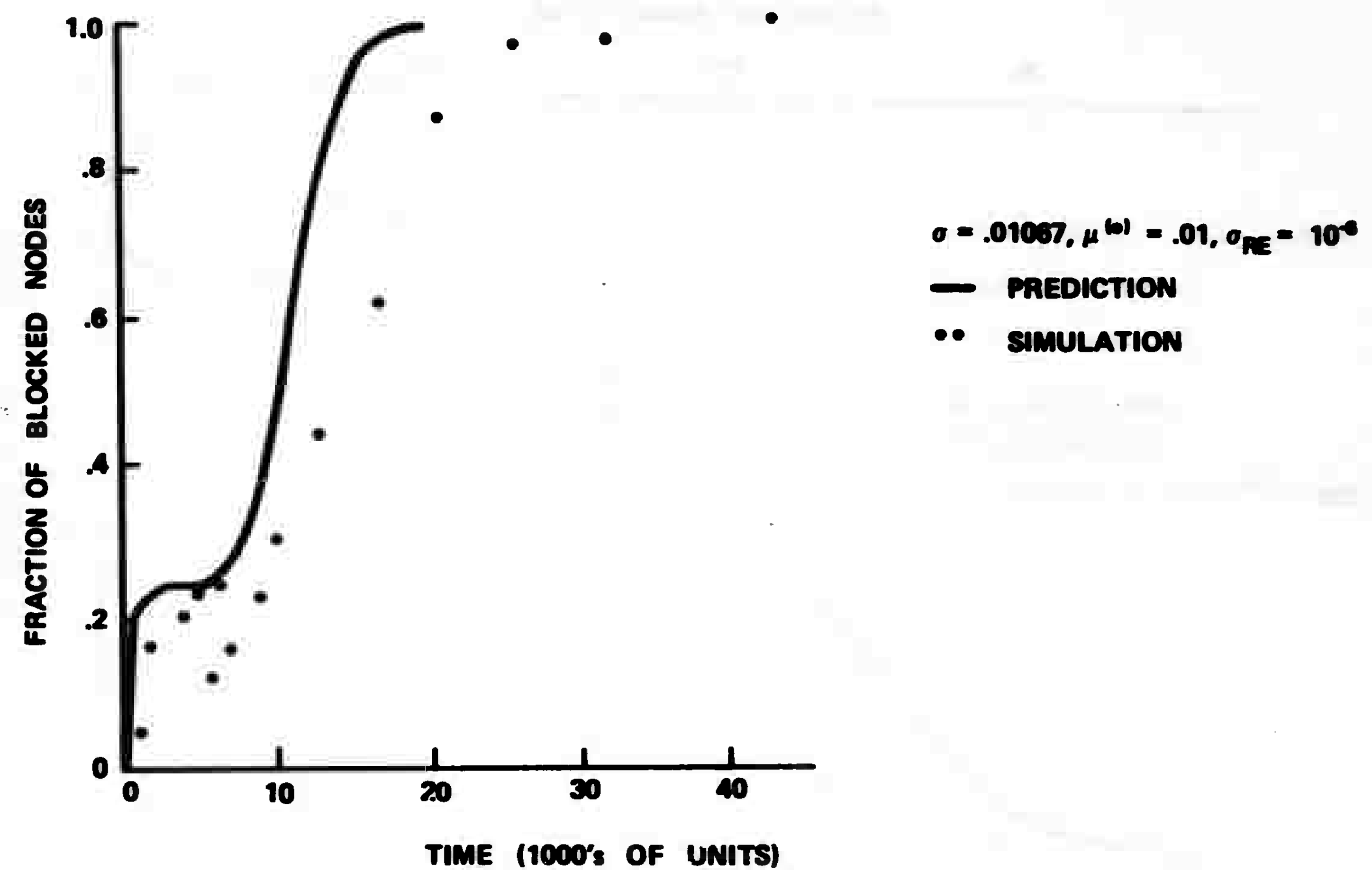


Figure 25. Store-and-Forward Network III

64,000 and 52,000 time units respectively. The predicted time to complete blocking from the modified network model with initial conditions $p_{40}(0) = 1$ and $f(0) = 0$ is 77,000 time units. In Fig. 25 the prediction from the modified network model assumes P[store-and-forward message arrival] = 1, and P[HOST arrival] = 0 and the same initial conditions as before. One of the reasons that the fit between the simulations and the predicted trajectories is not better is the difficulty of achieving uniform initial conditions for the simulated network, which are assumed in the modified network model.

Simulation results for the network with $\sigma = .02$, $\mu^{(0)} = .01$, and σ_{RE} equal successively to 1,000, .002, and 10^{-6} yielded net times to total blocking of 46,000; 22,000; and 24,000 time units respectively. The prediction from the modified network model is 18,000 time units.

We see that this model is far from being perfect, but it does provide nearly quantitative and certainly qualitative understanding of the behavior of these simulated networks.

CHAPTER 6

CONCLUSIONS

A number of new models that have application to store-and-forward communication networks have been presented.

First, we have the probabilistic model for nodal blocking due to finite storage space (Fig. 2 and Eqs. (3-7)). The model is applicable when the average message arrival rate σ equals or exceeds the average message service rate $\mu^{(0)}$. The model shows that the blocking behavior of an IMP is approximately a two-state Markov process.

Our second model is for the fraction of blocked nodes in a network of such nodes and also has a two-state Markov process representation (Fig. 4 and Eq. (18)). The result appears valid for both randomly connected and lattice networks and for a variety of system parameters (Figs. 9-14). However, the model for the fraction of blocked nodes in a "partial blocking" network (Figs. 15-17) needs to be greatly improved.

Various clumping models have been presented and shown useful for such a network. The clump size distribution for the case $\sigma = \mu^{(0)}$ (Eq. (30) and Fig. 18) and the maximum clump size model (Eq. (38) and Figs. 20 and 21) appear adequate to describe these cases. The average clump size for the case $\sigma > \mu^{(0)}$ (Eqs. (35-37)) is a fair approximation and needs further work.

The modified network model (Fig. 22) provides a clue to the fundamental behavior of store-and-forward communication networks that are subject to overutilization. The model treats the case $\sigma > \mu^{(0)}$ fairly

Preceding page blank

well (Figs. 23-25) but does not appear applicable in the marginal case $\sigma = \mu^{(0)}$.

Further work on models of this type appears justified. An effort should be made to improve the modified network model for the case $\sigma = \mu^{(0)}$, and investigations should be made into the transient clumping behavior in completely blocking networks. Also, the variance of the measurement of the fraction blocked in such networks, and the time dependent connectivity requires investigation.

Questions regarding the behavior of networks with selective blocking, as in the ARPA network, remain unanswered; nor have we introduced the effect of multi-packet messages. These would be important (and difficult) areas for research.

The whole subject of blocking in networks of this type appears to be absent from the literature. We believe that this field contains many additional challenging research areas.

BIBLIOGRAPHY

1. Roberts, L.G., and Wessler, B.D. "Computer network development to achieve resource sharing," AFIPS Conference Proceedings, Vol. 36, pp. 543-549, May 1970.

2. Heart, F.E., et al. "The interface message processor for the ARPA computer network," AFIPS Conference Proceedings, Vol. 36, pp. 551-567, May 1970.

3. Kleinrock, L. "Analytic and simulation methods in computer network design," AFIPS Conference Proceedings, Vol. 36, pp. 569-578, May 1970.

4. Frank, H., et al. "Topological considerations in the design of the ARPA computer network," AFIPS Conference Proceedings, Vol. 36, pp. 581-587, May 1970.

5. Carr, C.S., et al. "HOST-HOST communication protocol in the ARPA network," AFIPS Conference Proceedings, Vol. 36, pp. 589-597, May 1970.

6. Walker, P.M., and Mathison, S.L. "Communication carriers: evolution or revolution?," Technology Review, Vol. 73, pp. 44-55, October-November 1970.

7. Cox, D.R., and Smith, W.L. Queues, London, Methuen, 1968.

8. Rapoport, A. "Ignition phenomena in random nets," Bulletin of Mathematical Biophysics, Vol. 14, pp. 35-44, 1952.

9. Allanson, J.T. "Some properties of a randomly connected neural network," Symposium on Information Theory (C. Cherry, Ed.), London, Butterworths, 1955, pp. 303-313.

10. Broadbent, S.R., and Hammersley, J.M. "Percolation processes, crystals, and mazes," Proceedings Cambridge Philosophical Society, Vol. 53, pp. 629-641, 1957.

11. Gilbert, E.N. "Random plane networks," J. Society Indust. Applied Math., Vol. 9, pp. 533-543, December 1961.

12. Jacobs, I. "Connectivity of probabilistic graphs," MIT Research Lab. of Electronics, Cambridge, Mass., Tech. Report 356, September 15, 1959.

13. Frank, H., and Frisch, I.T. Communication, Transmission, and Transportation Networks, Reading, Mass., Addison-Wesley, 1971, Chap. 8.

14. Eden, M. "A two-dimensional growth process," Proceedings Fourth Berkeley Symposium on Math. Stat. and Prob., Vol. 4, pp. 223-239, 1961.

15. Morgan, R.W., and Welsch, D.J.A. "A two-dimensional Poisson growth process," J. Royal Statistical Society, Series B, Vol. 27, pp. 497-504, 1965.

16. Roach, S.A. The Theory of Random Clumping, London, Methuen, 1968.

17. Burke, P. J. "The output of a queueing system," Operations Research, Vol. 4, pp. 699-704, 1956.

18. Reich, E. "Departure Processes," Congestion Theory (W.L. Smith and W.E. Wilkinson, Eds.), Chapel Hill, N.C., Univ. of North Carolina Press, 1965, pp. 439-451.

19. Hildebrand, F.B. Advanced Calculus for Applications, Englewood Cliffs, N.J., Prentice-Hall, 1965, p. 150.

20. Cox, D.R. Renewal Theory, London, Methuen, 1962.

21. Bartlett, M. Stochastic Population Models in Ecology and Epidemiology, London, Methuen, 1960, Chap. 8.

22. Kendall, M.G., and Moran, P.A.P. Geometrical Probability, New York, Hafner Publishing Co., 1963, p. 69.

23. Cramer, H. Mathematical Methods of Statistics, Princeton, N.J., Princeton University Press, 1963, p. 194.

24. Feller, W. An Introduction to Probability Theory and Its Applications, Vol. 1, New York, Wiley, 1957, p. 343.

25. Grenander, U., and Szego, G. Toeplitz Forms and their Applications, Los Angeles, Univ. of California Press, 1958, p. 67.

26. Bellman, R.E. Introduction to Matrix Analysis, New York, McGraw-Hill, 1960, pp. 234-235.

27. Pritsker, A.A.B., and Kiviat, P.J. Simulation with GASP II, Englewood Cliffs, N.J., Prentice-Hall, 1969.

APPENDIX

A. Solution of $\dot{P} = AP + C$ for some special cases

In this section we solve the network equation

$$\dot{P} = AP + C$$

for some special network topologies. Recall that if there are m nodes in the net, then $P(t)$ is the $m \times 1$ matrix whose i^{th} component is the probability that node i is blocked at time t. A is an $m \times m$ constant matrix and C is an $m \times 1$ constant matrix. The solution is

$$P(t) = e^{At}P(0) + A^{-1}(e^{At} - I)C$$

Thus our problem is to find the exponential and the inverse of the matrix A.

1. Lattice

Consider a network consisting of $m = n^2$ nodes arranged in an $n \times n$ grid with 4-neighbor connections. For this system the matrix A is $n^2 \times n^2$ and takes the following form:

$$A = \begin{bmatrix} D & \Lambda & & & & \bigcirc \\ \Lambda & D & \Lambda & & & \\ & \Lambda & D & \Lambda & & \\ & & & \cdots & & \\ & \bigcirc & & & \Lambda & D \\ & & & & & \Lambda \quad D \end{bmatrix}$$

$$\text{where } D = \begin{bmatrix} a & b & & & & \\ b & a & b & & \bigcirc & \\ & b & a & b & & \\ & & & \cdots & & \\ & \bigcirc & & b & a & b \\ & & & & b & a \end{bmatrix} \quad n \times n$$

and

$$\Lambda = bI_n$$

where

$a = -\sigma$, $b = \frac{\mu^{(0)}}{5}$, and I_n is the $n \times n$ identity matrix.

We must first find the eigenvalues γ_v of D which are the solutions of $|D - \gamma I| = 0$. Let a stand for $a - \gamma$ in D; we wish to find the zeros of the determinant of D. Expanding by the elements of the top row, we note the following recurrence relation for the determinant Δ_n of the $n \times n$ matrix D:

$$\Delta_n = a\Delta_{n-1} - b^2\Delta_{n-2}$$

with initial conditions $\Delta_1 = a$, $\Delta_0 = 1$, $\Delta_{-1} = 0$. Following Grenander and Szego [25] we substitute $a = 2b \cos \theta$, assume a solution of the form $\Delta_n = \rho^n$, and solve the resulting quadratic in ρ. After satisfying the initial conditions the result is simply

$$\Delta_n = b^n \frac{\sin(n + 1)\theta}{\sin \theta}$$

which vanishes for

$$\theta = v\pi/n+1 \qquad v = 1,2, .., n$$

Therefore, the eigenvalues of D are

$$a - 2b \cos \frac{v\pi}{n+1} \qquad v = 1,2, .., n$$

which are all distinct. The eigenvectors are the solutions of

$$\begin{bmatrix} a & b & & & \bigcirc \\ b & a & b & & \\ & b & a & b & \\ & & & \cdots & \\ \bigcirc & & & b & a \end{bmatrix} \begin{bmatrix} X_{v1} \\ X_{v2} \\ X_{v3} \\ \cdots \\ X_{vn} \end{bmatrix} = \gamma_v \begin{bmatrix} X_{v1} \\ X_{v2} \\ X_{v3} \\ \cdots \\ X_{vn} \end{bmatrix}$$

It is easy to verify that the normalized solutions are

$$X_{vk} = \frac{(-1)^{n-k}}{\sqrt{\frac{n+1}{2}}} \sin \frac{kv\pi}{n+1}$$

so that the (i,j) element of e^D

$$e^D_{i,j} = \sum_{v=1}^{n} e^{\gamma_v} X_{vi} X_{vj}$$

and

$$D^{-1}_{i,j} = \sum_{v=1}^{n} (\gamma_v)^{-1} X_{vi} X_{vj} .$$

where

$$\gamma_v = a - 2b \cos \frac{v\pi}{n+1}$$

and

$$X_{vk} = \frac{(-1)^{n-k}}{\sqrt{\frac{n+1}{2}}} \sin \frac{kv\pi}{n+1}$$

Similarly, it is easy to show that the transformation R*AR (where R* is the transpose of R) where

$$R \equiv \begin{bmatrix} X_{11}I_n & \cdots & X_{v1}I_n & \cdots & X_{n1}I_n \\ X_{12}I_n & \cdots & X_{v2}I_n & \cdots & X_{n2}I_n \\ \\ X_{1n}I_n & \cdots & X_{vn}I_n & \cdots & X_{nn}I_n \end{bmatrix} \text{ with } X_{vk} \text{ as given}$$

above reduces Λ to the quasi-diagonal form

$$\begin{bmatrix} M_1 & & \bigcirc \\ & M_2 & \\ \bigcirc & \cdots & M_n \end{bmatrix}$$

where

$$M_v = D - 2b \cos \frac{v\pi}{n+1} I_n$$

Since M_v is equal to D with a change of the diagonal element, we have that the (k,l) element of the (i,j) block of e^A is

$$e^A_{i,j;k,l} = \sum_{v=1}^{n} X_{vi}X_{vj} \sum_{p=1}^{n} \exp(a-2b \cos\frac{v\pi}{n+1} -2b \cos\frac{p\pi}{n+1}) X_{pk}X_{pl}$$

and

$$A^{-1}_{i,j;k,l} = \sum_{n=1}^{n} X_{vi}X_{vj} \sum_{p=1}^{n} (a-2b \cos\frac{v\pi}{n+1} - 2b \cos\frac{p\pi}{n+1})^{-1} X_{pk}X_{pl}$$

where

$$X_{vk} = \frac{(-1)^{n-k}}{\sqrt{\frac{n+1}{2}}} \sin \frac{kv\pi}{n+1}$$

In our system $a = -\sigma$ and $b = \mu^{(0)}/5$ so the time constants, i.e., the arguments in each of the exponentials appearing in the solution for e^{At} are of the form

$$-\sigma t - \frac{2\mu^{(0)}}{5}\, t\left[\cos\frac{v_i\pi}{n+1} + \cos\frac{v_j\pi}{n+1}\right]$$

which takes on its smallest absolute value for $v_i = v_j = n$. Thus the motion of the system is bounded by

$$\exp - (\sigma - \frac{4}{5}\,\mu^{(0)}\cos\frac{\pi}{n+1})t$$

The number n is the square root of the number of nodes in the square lattice. This result shows that as $n \to \infty$ the system attains its steady state at a rate

$$\exp - (\sigma - \frac{4}{5}\,\mu^{(0)})t$$

which agrees with simulation results for $n = 32$ (see Eq. (18) and Figs. 9-11).

2. <u>Torus</u>

Again we consider a network of $m = n^2$ nodes arranged in an $n \times n$ grid with 4-neighbor connections, but this time we assume that opposite sides of the grid are connected together. The result is a torus, and for this case the matrix A, again $n^2 \times n^2$, takes the following form:

$$A = \begin{bmatrix} D & \Lambda & & & \Lambda \\ \Lambda & D & \Lambda & & \bigcirc \\ & & \cdots & & \\ & & \Lambda & D & \Lambda \\ \Lambda & \bigcirc & & \Lambda & D \end{bmatrix}$$

$$\text{where } D = \begin{bmatrix} a & b & & & b \\ b & a & b & & \bigcirc \\ & & \cdots & & \\ & \bigcirc & b & a & b \\ b & & & b & a \end{bmatrix}_{n \times n}$$

and

$$\Lambda = bI_n$$

where $a = -\sigma$, $b = \mu^{(0)}/5$, and I_n is the $n \times n$ identity matrix. The solution follows from the fact that A is a block circulant matrix [26].

It is easy to verify that the transformation R^*AR (where R^* is the transpose of R) where

$$R \equiv \begin{bmatrix} X_{10}I_n & & X_{i0}I_n & & X_{n0}I_n \\ X_{11}I_n & \cdots & X_{i1}I_n & \cdots & X_{n1}I_n \\ \vdots & & \vdots & & \vdots \\ X_{1n-1}I_n & & X_{in-1}I_n & & X_{nn-1}I_n \end{bmatrix}$$

and

$$X_{1k} = 1/\sqrt{n} \qquad k = 0,1, .., n-1$$

$$\left.\begin{aligned} X_{vk} &= \sqrt{\frac{2}{n}} \sin \frac{kv\pi}{n} \\ X_{v+1k} &= \sqrt{\frac{2}{n}} \cos \frac{kv\pi}{n} \end{aligned}\right\} \quad \begin{aligned} &v \text{ even}, \neq 0, < n \\ &k = 0,1, .., n-1 \end{aligned}$$

$$X_{nk} = \frac{(-1)^k}{\sqrt{n}} \quad \text{if } n \text{ even}; \ k = 0,1, .., n-1$$

reduces A to the quasi-diagonal form

$$\begin{bmatrix} M_1 & & & & & \\ & \ddots & & & \bigcirc & \\ & & M_v & & & \\ & & & M_{v+1} & & \\ & \bigcirc & & & \ddots & \\ & & & & & M_n \end{bmatrix}$$

where

$$M_1 = D + 2\Lambda$$

$$M_v = M_{v+1} = D + 2\Lambda \cos \frac{v\pi}{n} \qquad v \text{ even}, \neq 0, < n$$

$$M_n = D - 2\Lambda \qquad \text{if } n \text{ even}$$

Therefore, the (i,j) block of e^A is

$$e^A_{i,j} = \sum_{s=1}^{n} X_{si} X_{sj} e^{M_s}$$

and

$$A^{-1}_{i,j} = \sum_{s=1}^{n} X_{si} X_{sj} (M_s)^{-1}$$

with M_s and X_{sk} as given above.

We observe that the matrix D is simply matrix A wherein each block is of dimension one; therefore, the (k,l) element of e^D is

$$e^D_{k,\ell} = \sum_{p=1}^{n} X_{pk} X_{p\ell} e^{m_p}$$

and

$$D^{-1}_{k,\ell} = \sum_{p=1}^{n} X_{pk} X_{p\ell} (m_p)^{-1}$$

where

$$m_1 = a + 2b$$

$$m_v = m_{v+1} = a + 2b \cos \frac{v\pi}{n} \qquad v \text{ even}, \neq 0, < n$$

$$m_n = a - 2b \qquad \text{if } n \text{ even}$$

Since $\Lambda = bI_n$, each of the matrices M_s is equal to D with a change of the diagonal element. Hence the $(k,1)$ element of the (i,j) block of e^A is

$$e^A_{i,j;k,\ell} = \sum_{s=1}^{n} X_{si}X_{sj} \sum_{p=1}^{n} X_{pk}X_{p\ell}e^{m_{ps}}$$

and

$$A^{-1}_{i,j;k,\ell} = \sum_{s=1}^{n} X_{si}X_{sj} \sum_{p=1}^{n} X_{pk}X_{p\ell}(m_{ps})^{-1}$$

where

$$m_{1s} = a_s + 2b$$

$$m_{vs} = m_{v+1s} = a_s + 2b \cos \frac{v\pi}{n} \qquad v \text{ even}, \neq 0, < n$$

$$m_{ns} = a_s - 2b \qquad \text{if } n \text{ even}$$

and

$$a_1 = a + 2b$$

$$a_r = a_{r+1} = a + 2b \cos \frac{r\pi}{n} \qquad r \text{ even}, \neq 0, < n$$

$$a_n = a - 2b \qquad \text{if } n \text{ even}$$

and with x_{ij} as given before.

3. <u>Twisted Torus</u>

Once more we consider a network of $m = n^2$ nodes arranged in an

$n \times n$ grid with 4-neighbor connections. We assume that nodes are numbered from 1 to n^2 by rows and that node i is connected to nodes $i \pm 1$, $i \pm n$ modulo n^2. A "twisted torus" results for which the connection matrix A is as follows:

$$A = \begin{bmatrix} D & \Lambda & & & \Lambda^* \\ \Lambda^* & D & \Lambda & & \bigcirc \\ & & \cdots & & \\ & \bigcirc & \Lambda^* & D & \Lambda \\ \Lambda & & & \Lambda^* & D \end{bmatrix}$$

$$\text{where } D = \begin{bmatrix} a & b & & & \\ b & a & b & & \bigcirc \\ & & \cdots & & \\ & \bigcirc & b & a & b \\ & & & b & a \end{bmatrix}_{n \times n}$$

$$\text{and } \Lambda = \begin{bmatrix} b & & & \\ & b & \bigcirc & \\ & & \cdots & \\ \bigcirc & & b & \\ b & & & b \end{bmatrix}$$

The matrix is a circulant, i.e., any row is a cyclic shift of the previous row. Following Bellman [26] we find that the eigenvalues are

$$\gamma_k = a + b\left(e^{\frac{2\pi k i}{n^2}} + e^{\frac{2\pi k n i}{n^2}} + e^{\frac{2\pi k}{n^2}(n^2-n)i} + e^{\frac{2\pi k}{n^2}(n^2-1)i}\right)$$

$$= a + 2b\left(\cos\frac{2\pi k}{n^2} + \cos\frac{2\pi k}{n}\right) \qquad k = 0,1,..,n^2-1$$

where $i = \sqrt{-1}$, and with associated eigenvectors

$$\begin{bmatrix} 1 \\ e^{\frac{2\pi k}{n^2} i} \\ \vdots \\ e^{\frac{2\pi k}{n^2}(n^2-1) i} \end{bmatrix}$$

The eigenvalues occur in pairs in all but the extreme cases. This can be shown easily:

$$\gamma_{n^2-k} = a + 2b(\cos \frac{2\pi(n^2 - k)}{n^2} + \cos \frac{2\pi(n^2 - k)}{n})$$

$$= a + 2b(\cos \frac{2\pi k}{n^2} + \cos \frac{2\pi k}{n})$$

$$= \gamma_k$$

Thus the eigenvalues are

$$\gamma_1 = a + 4b$$

$$X_{1k} = 1/n$$

$$\left.\begin{aligned} X_{vk} &= \frac{\sqrt{2}}{n} \sin \frac{kv\pi}{n^2} \\ X_{v+1k} &= \frac{\sqrt{2}}{n} \cos \frac{kv\pi}{n^2} \end{aligned}\right\} \quad \begin{aligned} &v \text{ even}, \neq 0, < n^2 \\ &k = 0, 1, .., n^2-1 \end{aligned}$$

$$X_{n^2k} = \frac{(-1)^k}{n} \qquad n \text{ even}; k = 0, 1, .., n^2-1$$

Therefore, for the twisted torus

$$f(A)_{i,j} = \sum_{s=1}^{n^2} X_{si} X_{sj} f(\gamma_s)$$

where $f(y)_{i,j}$ is the i,j component of any power of y or its inverse, and X_{sk} and γ_s are as given above.

B. Clumping Analysis for 8-Neighbor Topologies (Assumed valid for $\sigma = \mu^{(0)}$, $N \geq 50$)

The different topologies for clumps of up to four blocked nodes with their corresponding birth and death rates are as follows:

1)
```
O O O
O X O
O O O
```
X = blocked node

O = free node

$\lambda_0 = \lambda^{(0)}$

$\lambda_1 = 8\lambda^{(1)}$

$\mu_1 = \mu^{(0)}$

2-1)
```
O O O O
O X X O
O O O O
```
$\lambda_2 = 4\lambda^{(2)} + 6\lambda^{(1)}$

$\mu_2 = 2\mu^{(1)}$

2-2)
```
  O O O
O O X O
O X O O
O O O
```
$\lambda_2 = 2\lambda^{(2)} + 10\lambda^{(1)}$

$\mu_2 = 2\mu^{(1)}$

3-1)
```
O O O O
O X X O
O O X O
  O O O
```
$\lambda_3 = \lambda^{(3)} + 4\lambda^{(2)} + 7\lambda^{(1)}$

$\mu_3 = 3\mu^{(2)}$

```
3-2)    O   O   O   O   O
        O   X   X   X   O
        O   O   O   O   O
```

$$\lambda_3 = 2\lambda^{(3)} + 4\lambda^{(2)} + 6\lambda^{(1)}$$
$$\mu_3 = \mu^{(2)} + 2\mu^{(1)}$$

```
3-3)            O   O   O
        O   O   O   X   O
        O   X   X   O   O
        O   O   O   O
```

$$\lambda_3 = \lambda^{(3)} + 4\lambda^{(2)} + 9\lambda^{(1)}$$
$$\mu_3 = \mu^{(2)} + 2\mu^{(1)}$$

```
3-4)            O   O   O
            O   O   X   O
        O   O   X   O   O
        O   X   O   O
        O   O   O
```

$$\lambda_3 = 4\lambda^{(2)} + 12\lambda^{(1)}$$
$$\mu_3 = \mu^{(2)} + 2\mu^{(1)}$$

```
3-5)    O   O   O   O   O
        O   X   O   X   O
        O   O   X   O   O
            O   O   O
```

$$\lambda_3 = \lambda^{(3)} + 3\lambda^{(2)} + 12\lambda^{(1)}$$
$$\mu_3 = \mu^{(2)} + 2\mu^{(1)}$$

```
4-1)    O   O   O   O
        O   X   X   O
        O   X   X   O
        O   O   O   O
```

$$\lambda_4 = 8\lambda^{(2)} + 4\lambda^{(1)}$$
$$\mu_4 = 4\mu^{(3)}$$

```
4-2)    O   O   O   O   O   O
        O   X   X   X   X   O
        O   O   O   O   O   O
```

$$\lambda_4 = 4\lambda^{(3)} + 4\lambda^{(2)} + 6\lambda^{(1)}$$
$$\mu_4 = 2\mu^{(2)} + 2\mu^{(1)}$$

```
4-3)      O  O  O  O
       O  O  X  X  O
       O  X  X  O  O
       O  O  O  O
```

$$\lambda_4 = 2\lambda^{(3)} + 4\lambda^{(2)} + 8\lambda^{(1)}$$
$$\mu_4 = 2\mu^{(3)} + 2\mu^{(2)}$$

```
4-4)         O  O  O
       O  O  O  X  O
       C  X  X  X  O
       O  O  O  O  O
```

$$\lambda_4 = \lambda^{(4)} + \lambda^{(3)} + 5\lambda^{(2)} + 7\lambda^{(1)}$$
$$\mu_4 = \mu^{(3)} + 2\mu^{(2)} + \mu^{(1)}$$

```
4-5)      O  O  O
       O  O  X  O  O
       O  X  X  X  O
       O  O  O  O  O
```

$$\lambda_4 = 3\lambda^{(3)} + 2\lambda^{(2)} + 9\lambda^{(1)}$$
$$\mu_4 = 2\mu^{(3)} + 2\mu^{(2)}$$

```
4-6)   O  O  O  O  O
       O  X  O  X  O
       O  X  X  O  O
       O  O  O  O
```

$$\lambda_4 = \lambda^{(4)} + 6\lambda^{(2)} + 8\lambda^{(1)}$$
$$\mu_4 = \mu^{(3)} + 2\mu^{(2)} + \mu^{(1)}$$

```
4-7)         O  O  O
          O  O  X  O
       O  O  X  O  O
       O  X  X  O
       O  O  O  O
```

$$\lambda_4 = 2\lambda^{(3)} + 4\lambda^{(2)} + 10\lambda^{(1)}$$
$$\mu_4 = \mu^{(3)} + 2\mu^{(2)} + \mu^{(1)}$$

```
4-8)   O  O  O  O  O  O
       O  X  O  O  X  O
       O  O  X  X  O  O
          O  O  O  O
```

$$\lambda_4 = 2\lambda^{(3)} + 4\lambda^{(2)} + 12\lambda^{(1)}$$
$$\mu_4 = 2\mu^{(2)} + 2\mu^{(1)}$$

```
4-9)   O  O  O  O  O
       O  X  O  X  O  O
       O  O  X  O  X  O
          O  O  O  O  O
```

$$\lambda_4 = 2\lambda^{(3)} + 4\lambda^{(2)} + 12\lambda^{(1)}$$
$$\mu_4 = 2\mu^{(2)} + 2\mu^{(1)}$$

```
4-10)     O  O  O
          O  X  O
       O  O  X  O  O
       O  X  O  X  O
       O  O  O  O  O
```

$$\lambda_4 = 3\lambda^{(3)} + 3\lambda^{(2)} + 11\lambda^{(1)}$$
$$\mu_4 = \mu^{(3)} + 3\mu^{(1)}$$

```
4-11)  O  O  O
       O  X  O  O  O
       O  O  X  X  O
          O  X  O  O
          O  O  O
```

$$\lambda_4 = 3\lambda^{(3)} + 2\lambda^{(2)} + 11\lambda^{(1)}$$
$$\mu_4 = \mu^{(3)} + 2\mu^{(2)} + \mu^{(1)}$$

The approximation for large n is again a straight line topology. In simulations on an 8-neighbor lattice round clumps are observed more frequently than stringy ones, but for clump topologies up to 4, the straight line clump yields better approximate growth and death rates than does, say, a square clump. The approximation gives

$$\lambda_n = 2(n-2)\lambda^{(3)} + 4\lambda^{(2)} + 6\lambda^{(1)} = (6n+2)\lambda^{(1)}$$

$$\mu_n = (n-2)\mu^{(2)} + 2\mu^{(1)} = n\mu^{(0)} - \frac{(2n-2)}{9}\mu^{(0)}$$

$$= \frac{7n+2}{9}\mu^{(0)}$$

We find that a better approximation for $n = 4$ (and thus we will assume for larger clumps as well) is

$$\lambda_n = 6n\,\lambda^{(1)}$$

$$\mu_n = \frac{7n}{9}\mu^{(0)}$$

For the stationary probability of a clump of size n we then get

$$p_n = p_0 \frac{\lambda^{(0)}}{\mu^{(0)}} \frac{r^{n-1}}{n} \qquad n \geq 1$$

and

$$p_0 = \left[1 - \frac{\lambda^{(0)}}{\mu^{(0)}} \frac{\ln(1-r)}{r}\right]^{-1}$$

where

$$r = \frac{54}{7} \frac{\lambda^{(1)}}{\mu^{(0)}}$$

$$E[\#\text{ in system}] = \frac{\lambda^{(0)}}{\mu^{(0)}} \frac{p_0}{(1-r)}$$

For the case $\sigma = \mu^{(0)}$, $N = 50$ this yields

$$E[\#\text{ in system}] = .1338$$

The result observed in simulations is about .10.

C. Simulation Programs

The following programs performed the simulations described earlier. They are written in Fortran IVH and run on the XDS Sigma-7 computer at U.C.L.A.

1. Lattice (with graphical display)
2. Random Graph
3. Message Transfer Network

Lattice

```
C     MAIN PROGRAM
      COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NBON, NRSEC(16), PCSEC(16),
     2   NRAND(128), LRAND, MRAND, KRAND, KGOOD, KBAD, PARAM2(10,10)
      COMMON GCODE(5000), JAR(32,32), INT1(2), INT2(2)
      DIMENSION JINT1(2), JINT2(2)
      INTEGER ATRIB, TNOW, TFIN
      INTEGER GCODE
      DATA IGOOD, IBAD/' ','8'/
      KGOOD=IGOOD
      KBAD=IBAD
      CALL DSOPEN(GCODE,5000)
      DATA JINT1(1),JINT1(2)/'0800','100A'/
      DATA JINT2(1),JINT2(2)/'0800','100F'/
      INT1(1)=JINT1(1)
      INT1(2)=JINT1(2)
      INT2(1)=JINT2(1)
      INT2(2)=JINT2(2)
      CALL BEGPIC('PICTURE')
C   BEGIN PICTURE DEFINITION
      IX=153
      IY=950
      DO 300 I=1,32
      J=1
      K=INTENS(2)
      CALL DDSTP(IPT)
C   IPT IS THE CURRENT STACK POINTER
C   INTENSITY = 2 IS INTENSITY OF GOOD NODES
C   INTENSITY = 7 IS INTENSITY OF BAD NODES
      JAR(I,J)=IPT
C   JAR(I,J)=ABS ADDRESS OF INTENSITY INSTR. FOR POINT (I,J)
      K=POINT(IX,IY)
      DO 200 J=2,32
      K=INTENS(2)
      CALL DDSTP(IPT)
      JAR(I,J)=IPT
      K=RPOINT(24,0)
  200 CONTINUE
      IX=153
      IY=IY-24
  300 CONTINUE
      CALL ENDPIC
      CALL DDPAD('PICTURE',IBP)
      DO 31 I=1,32
      DO 31 J=1,32
   31 JAR(I,J)=JAR(I,J)-IBP+1
C   JAR(I,J)=REL ADDRESS OF INTENSITY INSTR. FOR POINT (I,J)
      CALL DISPIC
C   DISPLAY PICTURE
      READ(105,143) LRAND,MRAND,KRAND,NRAND(1)
  143 FORMAT(4I9)
      DO 144 J=2,128
  144 NRAND(J)=NRAND(J-1)+2*J
      JRUN=1
    5 CALL DATAN
      CALL GASP
      DO 3 I=1,32
      DO 3 J=1,32
    3 CALL DDREP('PICTURE',JAR(I,J),INT1)
      IF (JRUN .GE. 10)  GO TO 57
C   LAST RUN IS NO 10.  THIS CARD CAN BE ALTERED TO ALLOW ANY NUMBER
C         OF RUNS UP  TO 10 WITH DIFFERENT PARAMETERS FOR EACH RUN.
      JRUN=JRUN+1
      GO TO 5
   57 STOP
      END
C
C
      SUBROUTINE DRAND (RNUM)
      COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISFED, TNOW, NBDN, NBSEC(16), PCSEC(16),
     2   NRAND(128), LRAND, MRAND, KRAND
      LRAND=LRAND*65539
      MRAND=MRAND*33554433
      J=1+IABS(LRAND)/16777216
      RNUM= .5+FLOAT(NRAND(J)+LRAND+MRAND)* .2328306E-9
      KRAND=KRAND*362436069
      NRAND(J)=KRAND
      RETURN
      END
C
C
      SUBROUTINE ORDER (M,N)
      COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISFED, TNOW, NBDN, NBSEC(16), PCSEC(16)
      INTEGER  AT5, AT6, AT7, AT8
      INTEGER ATRIB, TNOW, TFIN
C   ORDER ADDS NODE (M,N) TO ORDERED LIST OF EVENT TIMES FOR DATAN
      M5=0
      M7=0
C   M5 AND M7 ARE FLAGS. WHEN BOTH=1 WE'VE FOUND RIGHT SPOT FOR (M,N)
      I=1
      J=1
C   KNOW THAT NODE (1,1) HAS BEEN ORDERED
      IF (KRITE .EQ. 8)  GO TO 14
      I=ATRIB(5)
      J=ATRIB(6)
      IF (J .NE. 7777)  GO TO 14
      I=ATRIB(7)
      J=ATRIB(8)
   14 IF (ATRIB(1)-NSET(I,J,1)) 5,6,7
C     .LT. 0 MEANS GO TO PREDECESSOR,   .GT. 0 GO TO SUCCESSOR
    5 IA=NSET(I,J,7)
      IF (M7 .EQ. 1)  GO TO 9
      IF (IA .EQ. 9999) GO TO 9
C    SEE IF IT HAS A PREDECESSOR
      J=NSET(I,J,8)
      I=IA
      M5=1
      GO TO 14
    7 IA=NSET(I,J,5)
      IF (M5 .EQ. 1) GO TO 6
      IF (IA .EQ. 7777) GO TO 6
C    SEE IF IT HAS A SUCCESSOR
      J=NSET(I,J,6)
      I=IA
      M7=1
      GO TO 14
    6 ATRIB(5)=NSET(I,J,5)
C    (M,N) SUCCEDES (I,J) (BY CONVENTION IF THEY HAVE = TIMES)
      ATRIB(6)=NSET(I,J,6)
      ATRIB(7)=I
      ATRIB(8)=J
      NSET(I,J,5)=M
      NSET(I,J,6)=N
      AT5=ATRIB(5)
      IF (AT5 .EQ. 7777) GO TO 98
C    TEST FAILS IF (I,J) HAS A SUCCESSOR, AND THUS MUST UPDATE HIM
      AT6=ATRIB(6)
      NSET(AT5,AT6,7)=M
      NSET(AT5,AT6,8)=N
      GO TO 98
    9 ATRIB(5)=I
C    (M,N) PRECEDES (I,J)
      ATRIB(6)=J
      ATRIB(7)=NSET(I,J,7)
      ATRIB(8)=NSET(I,J,8)
      NSET(I,J,7)=M
      NSET(I,J,8)=N
      AT7=ATRIB(7)
      IF (AT7 .EQ. 9999) GO TO 98
C    TEST FAILS IF (I,J) HAS A PREDECESSOR (WHICH MUST BE UPDATED)
      AT8=ATRIB(8)
      NSET(AT7,AT8,5)=M
      NSET(AT7,AT8,6)=N
   98 DO 99 K=5,8
   99 NSET(M,N,K)=ATRIB(K)
      RETURN
      END
C
```

```
C
      SUBROUTINE RACK(J,K)
      COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NBDN, NBSEC(16), PCSEC(16),
     2   NRAND(128), LRAND, MRAND, KRAND, KGOOD, KBAD, PARAM2(10,10)
      COMMON GCODE(5000), JAR(32,32), INT1(2), INT2(2)
      INTEGER ATRIB, TNOW, TFIN
      INTEGER GCODE
C    (J,K) IS AN INITIALLY BAD NODE
      NSET(J,K,2)=1
      CALL DDREP('PICTURE',JAR(J,K),INT2)
C    WILL UPDATE NO OF BAD NODES IN SECTOR, AND PCT BAD IN SECTOR
      NSECT=NSET(J,K,3)
      NBSEC(NSECT)=NBSEC(NSECT)+1
      RSEC=NBSEC(NSECT)
      PCSEC(NSECT)=RSEC/64.0
      IF (J-32) 2,3,2
    2 NSET(J+1,K,4)=NSET(J+1,K,4)+1
    3 IF (J-1) 4,5,4
    4 NSET(J-1,K,4)=NSET(J-1,K,4)+1
    5 IF  (K-32) 6,7,6
    6 NSET(J,K+1,4)=NSET(J,K+1,4)+1
    7 IF (K-1) 8,9,8
    8 NSET(J,K-1,4)=NSET(J,K-1,4)+1
    9 RETURN
      END
C
C
      SUBROUTINE FIND (NROW,NCOL,NCODE)
      COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NBDN, NBSEC(16), PCSEC(16)
      INTEGER ATRIB, TNOW, TFIN, ALT
      IF ((NROW.LT.1.OR.NROW.GT.32).OR.(NCOL.LT.1.OR.NCOL.GT.32))
     1   GO TO 8
      JROW=NROW
      JCOL=NCOL
      ATRIB(1)=NSET(JROW,JCOL,1)
      ATRIB(2)=NSET(JROW,JCOL,2)
      ATRIB(4)=NSET(JROW,JCOL,4)
      JTIME=ATRIB(1)
      JNEHB=ATRIB(4)+1
      IF (NCODE .EQ. 0) ATRIB(4)=ATRIB(4)+1
      IF (NCODE .EQ. 1) ATRIB(4)=ATRIB(4)-1
      NSET(JROW,JCOL,4)=ATRIB(4)
      KNEHB=ATRIB(4)+1
      JSUB=5*ATRIB(2)+JNEHB
      KSUB=5*ATRIB(2)+KNEHB
      CALL CAF(JROW,JCOL,ALAMD)
      CALL URAND(RNUM)
      ALT=TNOW-INT(ALAMD*ALOG(RNUM)-0.5)
    3 ATRIB(1)=ALT
      GO TO 6
    4 IF(ATRIB(1)-ALT) 5,6,7
    5 IF (NCODE.NE.ATRIB(2)) ATRIB(1)=ALT
      GO TO 6
    7 IF (NCODE.EQ.ATRIB(2)) ATRIB(1)=ALT
    6 CONTINUE
      NSET(JROW,JCOL,1)=ATRIB(1)
      IF (JTIME.EQ.ATRIB(1))  GO TO 10
      CALL RMOVE(JROW,JCOL)
      CALL ORDER(JROW,JCOL)
   10 CONTINUE
    8 RETURN
      END
C
C
      SUBROUTINE RMOVE (JROW, JCOL)
      COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISFED, TNOW, NBDN, NHSEC(16), PCSEC(16)
      INTEGER ATRIB, TNOW, TFIN
      DO 10 K=1,8
   10 ATRIB(K)= NSET(JROW,JCOL,K)
      ISUCR=ATRIB(5)
      ISUCC=ATRIB(6)
      IPRER=ATRIB(7)
      IPREC=ATRIB(8)
      IF (IPRER .EQ. 9999) GO TO 15
      NSET(IPRER,IPREC,5)=ISUCR
      NSET(IPRER,IPREC,6)=ISUCC
      IF (ISUCR .EQ. 7777)  GO TO 25
   15 NSET(ISUCR,ISUCC,7)=IPRER
      NSET(ISUCR,ISUCC,8)=IPREC
   25 RETURN
      END
C
C
      SUBROUTINE GASP
      COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISFED, TNOW, NBDN, NHSEC(16), PCSEC(16)
      INTEGER ATRIB, TNOW, TFIN, SUCR, SUCC
      NEXTR=1
      NEXTC=1
      JRITE=0
    4 IF (NSET(NEXTR,NEXTC,7) .EQ. 9999)  GO TO 5
C   TEST ABOVE FAILS IF (NEXTR,NEXTC) IS NOT THE HEAD OF THE LIST
      NEX=NSET(NEXTR,NEXTC,7)
      NEXTC=NSET(NEXTR,NEXTC,8)
      NEXTR=NEX
C   SEE IF HIS PREDECESSOR IS HEAD OF LIST
      GO TO 4
    5   CALL RMOVE(NEXTR,NEXTC)
        ITEST=ISITON(8)
        IF(ITEST.NE.8) GO TO 598
  597   ITEST=ISITON(8)
        IF(ITEST.EQ.0)  RETURN
        GO TO 597
  598   CONTINUE
        TNOW=ATRIB(1)
        IF (TNOW .GE. TFIN(JRUN)) GO TO 17
        SUCR=ATRIB(5)
        SUCC=ATRIB(6)
        JRITE=JRITE+1
        KRITE=0
        IF (JRITE .LT. JRATE(JRUN)) GO TO 7
C   IF ABOVE TEST IS MET, WILL NOT PRINT RESULTS AT THIS EVENT TIME
        JRITE=0
        KRITE=1
C   WILL PRINT RESULTS AT THIS EVENT TIME
    7   CALL EVNTS(NEXTR,NEXTC)
        NEXTR=SUCR
        NEXTC=SUCC
        GO TO 4
   17   KRITE=1
        WRITE (108,21)
   21   FORMAT (1H0,52X,26H** FINAL REPORT FOLLOWS **)
        CALL EVNTS (NEXTR,NEXTC)
        RETURN
        END
C
C
        SUBROUTINE DATAN
        COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
       1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NBDV, NHSEC(16), PCSEC(16),
       2   NRAND(128), IRAND, MRAND, KRAND, KGOOD, KBAD, PARAM2(10,10)
        DIMENSION  MAP(64), BAD(20)
        INTEGER ATRIB, TNOW, TFIN, BAD
        REAL LAMBDA, MU
        IF ( JRUN .NE. 1)  GO TO 29
        READ (105,31) KRITE
   31   FORMAT (I1)
        IF (KRITE) 32,32,33
   32   DO 19 J=1,10
        READ (105,21)  LAMBDA, MU, N
        WRITE (108,10)  LAMBDA, MU, N
        IF (LAMBDA-MU)  6,7,8
    6   WRITE (108,9)
    9   FORMAT (50H0ERROR- LAMBDA MUST BE GREATER THAN OR EQUAL TO MU)
        JRUN=20
        RETURN
    7   P=N
```

```
      PARAM(1,J)=P/MU
      GØ TØ 11
    8 PARAM(1,J)=1.0/(LAMBDA-MU)
   11 PARAM(6,J)=1.0/MU
      DELTA=MU/5.0
      DØ 19 I=2,5
      MU=MU-DELTA
      PARAM(I,J)=1.0/(LAMBDA-MU)
   19 PARAM(I+5,J)=1.0/MU
   10 FØRMAT (8H0LAMBDA=,F7.4,5X,3HMU=,F7.4,5X,
     1     22HSIZE ØF QUEUEING RØØM=,I5)
   21 FØRMAT (2(F7.0),I5)
   33 DØ 34 I=1,10
   34 READ (105,55) (PARAM2(J,I),J=1,10)
   55 FØRMAT (10E7.2)
      DØ 37 I=1,10
      DØ 37 J=1,10
   37 PARAM2(I,J)=1./PARAM2(I,J)
   35 FØRMAT(10F7.0)
   36 READ (105,22) (JRATE(I),I=1,10)
   22 FØRMAT (10I7)
      READ (105,23) (TFIN(I),I=1,10)
   23 FØRMAT (10I7)
      DØ 4 J=1,10
      WRITE (108,3)  J, JRATE(J), TFIN(J)
    4 WRITE (108,5)  (PARAM(I,J),I=1,10)
    5 FØRMAT (1H ,11HPARAMETERS=,10(F7.4,5X))
    3 FØRMAT (1H0,7HRUN NØ.,I2,10X,12HREPØRT RATE=,I7,10X,
     1  12HFINISH TIME=,I7)
   29 READ (105,25) NIBN
      ISEED=0
      KRITE=8
   25 FØRMAT (I4)
C   NIBN IS NØ ØF INITIALLY BAD NØDES
      NBDN=NIBN
      READ (105,38) KØUNT(1)
   38 FØRMAT(I1)
      WRITE (108,26) JRUN, NIBN, KØUNT(1), JRATE(JRUN), TFIN(JRUN),
     1                 LRAND,MRAND,KRAND,NRAND(1)
   26 FØRMAT (1H1,30X,7HRUN NØ.,I2,10X,27HNØ. ØF INITIALLY BAD NØDES=,
     1  I4,10X,12HFIND ØPTIØN=,I1/1H0,4X,12HREPØRT RATE=,I7,10X,
     2  12HFINISH TIME=,I7,10X,13HRANDØM SEEDS=,4(I9,1X))
      WRITE (108,5)  (PARAM(I,JRUN),I=1,10)
C   WILL NØW SET SECTIØN NØS AND RESET ØTHER ELEMENTS ØF NSET
      LN=0
      DØ 14 ML=1,25,8
      ML7=ML+7
      DØ 14 I=1,25,8
      I7=I+7
      LN=LN+1
```

```
C   LN WILL BE THE SECTION NO
      DO 14 N=ML,ML7
      DO 14 J=1,17
      DO 13 K=1,2
   13 NSET(N,J,K)=0
      NSET(N,J,3)=LN
      DO 14 M=4,8
   14 NSET(N,J,M)=0
      DO 17 I=1,16
      NBSEC(I)=0
   17 PCSEC(I)=0
      READ (105,47) IDATA
   47 FORMAT (I1)
      IF (IDATA .EQ. 1) GO TO 99
C   IDATA=1 IF USING ALTERNATE FORM OF INPUT OF INITIALLY BAD NODES
      IF (NBDN .EQ. 0)  GO TO 44
      WRITE (108,53)
   53 FORMAT (1H0,19X,33HLIST OF INITIAL BAD NODES FOLLOWS)
      DO 43 I=1,NBDN,10
      READ (105,45) (BAD(J),J=1,20)
   45 FORMAT (20(I2))
      WRITE (108,54)  (BAD(J),J=1,20)
   54 FORMAT(1H0,10(2H (,I2,1H,,I2,2H) ))
      DO 43 M=1,10
      J=BAD(2*M-1)
      K=BAD(2*M)
C   BAD NODE IS (J,K)
      IF (J .EQ. 44) GO TO 44
C   J=44 MARKS END OF LIST OF INITIAL BAD NODES
      CALL BACK (J,K)
   43 CONTINUE
   44 DO 77 M=1,32
      DO 77 N=1,32
      JEVNT=NSET(M,N,2)
      NEWBD=NSET(M,N,4)
      JSUB=5*JEVNT+NEWBD+1
      CALL CAP(M,N,ALAMD)
      CALL DRAND(RNUM)
      NSET(M,N,1)=-INT(ALAMD*ALOG(RNUM)-0.5)
   77 CONTINUE
      IF (NSET(1,1,1)-NSET(1,2,1)) 78,76,79
   76 NSET(1,1,5)=1
      NSET(1,1,6)=2
      NSET(1,1,7)=9999
      NSET(1,1,8)=9999
      NSET(1,2,5)=7777
      NSET(1,2,6)=7777
      NSET(1,2,7)=1
      NSET(1,2,8)=1
      GO TO 63
   79 NSET(1,1,5)=7777
      NSET(1,1,6)=7777
      NSET(1,1,7)=1
      NSET(1,1,8)=2
      NSET(1,2,5)=1
      NSET(1,2,6)=1
      NSET(1,2,7)=9999
      NSET(1,2,8)=9999
   83 ML=0
      DO 80 I=1,32
      II=I
      DO 80 J=1,32
      JJ=J
      ML=ML+1
      IF (ML .LE. 2) GO TO 80
      ATRIB(1)=NSET(II,JJ,1)
      CALL ORDER(II,JJ)
   80 CONTINUE
      KRITE=0
      RETURN
   99 WRITE (108,101)
  101 FORMAT(1H0/1H0,16X,33HINITIAL GRID (8'S MARK BAD NODES)/1H0/1H0)
      DO 127 I=1,32,2
      READ (105,121) (MAP(J),J=1,64)
  121 FORMAT (64I1)
C   TWO ROWS OF THE GRID COMPRISE ONE LINE OF DATA
      WRITE (108,122) (MAP(J),J=1,32)
  122 FORMAT (1H ,32(I1,1X))
C   BAD NODES = 8, GOOD NODES = 1
      WRITE(108,122) (MAP(J),J=33,64)
      DO 123 M=1,32
      IF (MAP(M) .EQ. 1) GO TO 123
      J=I
      K=M
      CALL BACK (J,K)
  123 CONTINUE
      DO 127 M=33,64
      IF (MAP(M) .EQ. 1) GO TO 127
      J=I+1
      K=M-32
      CALL BACK (J,K)
  127 CONTINUE
      GO TO 44
C   JOINS PROGRAM WHERE REGULAR FORM OF INPUT ENDED
      END
C
C
      SUBROUTINE CNVRT(N,IROW,ICOL)
      DO 10 I=1,32
      IF (N-32) 5,5,10
   10 N=N-32
    5 INEW=N
      ICEL=I
      RETURN
      END
C
C
      SUBRØUTINE EVNTS (I,J)
      CØMMØN NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KØUNT(10), ISFED, TNØW, NØDN, NBSEC(16), PCSEC(16),
     2   NRAND(128), IRAND, MRAND, KRAND, KGØØD, KBAD, PARAM2(10,10)
      CØMMØN GCØDE(5000), JAR(32,32), INT1(2), INT2(2)
      DIMENSIØN MAP(32)
      INTEGER ATRIB, TNØW, TFIN
      INTEGER GCØDE
      II=I
      JJ=J
      JEVNT=ATRIB(2)
      IZ=JEVNT
      IF (JEVNT .EQ. 0)  ATRIB(2)=1
      IF (JEVNT .EQ. 1)  ATRIB(2)=0
      NSET(II,JJ,2)=ATRIB(2)
      KEVNT=ATRIB(2)
      NMBRW=I+1
      NMBCL=J
      CALL FIND (NMBRW,NMBCL,IZ)
      NMBRW=I-1
      CALL FIND (NMBRW,NMBCL,IZ)
      NMBRW=I
      NMBCL=J+1
      CALL FIND (NMBRW,NMBCL,IZ)
      NMBCL=J-1
      CALL FIND (NMBRW,NMBCL,IZ)
      DØ 1 L1=1,8
    1 ATRIB(L1)=NSET(II,JJ,L1)
      NEHBD=ATRIB(4)+1
      NSECT=ATRIB(3)
      JSUB=5*JEVNT+NEHBD
      KSUB=5*KEVNT+NEHBD
      CALL CAP(II,JJ,ALAMD)
C   ALAMD IS THE MEAN INTERARRIVAL TIME I.E. 1/LAMBDA ØR 1/MU
      CALL URAND(RNUM)
      ATRIB(1)=TNØW-INT(ALAMD*ALØG(RNUM)-0.5)
C   JEVNT=0, KEVNT=1 IF NØDE CHANGED FRØM GØØD TØ BAD, AND VICE VERSA
      ATRIB(3)=NSECT
      ATRIB(4)=NEHBD-1
      IF (JEVNT .EQ. 1)  GØ TØ 10
      CALL UDREP('PICTURE',JAR(II,JJ),INT2)
C   UPDATES INTENSITY ØF PØINT(I,J) IN DISPLAY
C   JAR(I,J)=REL ADDRESS ØF INTENSITY INSTR. FØR PØINT (I,J)
```

```
      NBDN=NBDN+1
C   NBDN IS  NO OF BAD NODES IN GRID
      NBSEC(NSECT)=NBSEC(NSECT)+1
C   NBSEC IS NO OF BAD NODES IN SECTION
      GO TO 12
   1u NBDN=NBDN-1
      NBSEC(NSECT)=NBSEC(NSECT)-1
      CALL DDREP('PICTURE',JAR(II,JJ),INT1)
   12 RSEC=NBSEC(NSECT)
      PCSEC(NSECT)=RSEC/64.0
      DO 99 I=1,?
   99 NSET(II,JJ,I)=ATRIB(I)
      IF (KRITE .EQ. 0)  GO TO 57
      RSEC=NBDN
      PCTNB=RSEC/1024.0
      WRITE(108,52) TNOW, NBDN, PCTNB
   52 FORMAT (1H0/1H0,5X,5HTIME=,I5,5X,17HNO. OF BAD NODES=,
     1      I4,5X,9HPCT. BAD=,F6.4)
      DO 18 MR=1,32
      DO 11 MC=1,32
      IF (NSET(MR,MC,2)) 14,15,14
   14 MAP(MC)=KBAD
      GO TO 11
   15 MAP(MC)=KGOOD
   11 CONTINUE
      WRITE (108,21) (MAP(MX),MX=1,32)
   18 CONTINUE
   21 FORMAT (1H ,32(A1,1X))
   57 CALL ORDER(II,JJ)
   19 RETURN
      END
C
C
      SUBROUTINE CAP(I,J,ALAMD)
      COMMON NSET(32,32,8), PARAM(10,10),JRATE(10), TFIN(10), ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISFED, TNOW, NBDN, NBSEC(16), PCSEC(16),
     2   NRAND(128), IRAND, MRAND, KRAND, KGOOD, KBAD, PARAM2(10,10)
      INTEGER ATRIB, TNOW, TFIN
      NCODE=5*NSET(I,J,2)+NSET(I,J,4)+1
      IF (I.EQ.16.AND.J.EQ.16)  GO TO 30
      IF (I.EQ.15.AND.J.EQ.16)  GO TO 30
      IF (I.EQ.14.AND.J.EQ.16)  GO TO 30
      ALAMD=PARAM2(NCODE,JRUN)
      RETURN
   30 CONTINUE
      ALAMD=PARAM(NCODE,JRUN)
      RETURN
      END
    SYMBOL
          DEF        ISITON
```

```
TSITAN    CAL2,1    0
          WD,0      0
          STCF      3
          CAL2,1    1
          SCS,3     4
          LW,4      13
          LW,13     1,4
          AND,3     *13
          B         2,4
          END
```

Random Graph

```
C       MAIN PROGRAM
        COMMON/7/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
       1 KRITE, JRUN, KOUNT(10), ISFED, TNOW, NBDN, NBSEC(16), PCSEC(16),
       2   NRAND(128), LRAND, MRAND, KRAND
        DIMENSION NROW(8), NCOL(8)
        INTEGER ATRIB, TNOW, TFIN
        READ(105,143) LRAND,MRAND,KRAND,NRAND(1)
  143   FORMAT(4I9)
        DO 144 J=2,128
  144   NRAND(J)=NRAND(J-1)+2*J
        CALL GRAPH
        READ (105,21) IGRAF
  21    FORMAT(I1)
        IF (IGRAF.EQ.0)  GO TO 95
        WRITE(108,37)
  37    FORMAT(1H1,2(4X,4HNODE,15X,9HNEIGHBORS,23X))
        DO 94 I=1,32
        DO 94 J=1,32,2
        JEX=J+1
        DO 4 K=9,12
        ME=NSET(I,J,K)
        CALL CNVRT(ME,JROW,JCOL)
        NROW(K-8)=JROW
        NCOL(K-8)=JCOL
        ME=NSET(I,JEX,K)
        CALL CNVRT(ME,JROW,JCOL)
        NROW(K-4)=JROW
  4     NCOL(K-4)=JCOL
        M=JEX
        WRITE(108,17) I,J,NROW(1),NCOL(1),NROW(2),NCOL(2),NROW(3),NCOL(3),
       1  NROW(4),NCOL(4),
       2              I,M,NROW(5),NCOL(5),NROW(6),NCOL(6),NROW(7),NCOL(7),
       3  NROW(8),NCOL(8)
  17    FORMAT(1H ,2(5(3H  (,I2,1H,,I2,1H)),10X))
  94    CONTINUE
  95    CONTINUE
        JRUN=1
  5     CALL DATAN
        CALL GASP
        IF (JRUN .GE. 10)  GO TO 57
C   LAST RUN IS NO 10.  THIS CARD CAN BE ALTERED TO ALLOW ANY NUMBER
C       OF RUNS UP  TO 10 WITH DIFFERENT PARAMETERS FOR EACH RUN.
        JRUN=JRUN+1
        GO TO 5
  57    STOP
        END
C
C
        SUBROUTINE DRAND (RNUM)
        COMMON//NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NODN, NOSEC(16), PCSEC(16),
     2   NRAND(128), LRAND, MRAND, KRAND
      LRAND=LRAND*65539
      MRAND=MRAND*33554433
      J=1+IABS(LRAND)/16777216
      RNUM= .5+FLOAT(NRAND(J)+LRAND+MRAND)* .23283064E-9
      KRAND=KRAND*362436069
      NRAND(J)=KRAND
      RETURN
      END
C
C
      SUBROUTINE ORDER (M,N)
      COMMON//NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NODN, NOSEC(16), PCSEC(16)
      INTEGER  AT5, AT6, AT7, AT8
      INTEGER ATRIB, TNOW, TFIN
C   ORDER ADDS NODE (M,N) TO ORDERED LIST OF EVENT TIMES FOR DATAN
      M5=0
      M7=0
C   M5 AND M7 ARE FLAGS, WHEN BOTH=1 WE'VE FOUND RIGHT SPOT FOR (M,N)
      I=1
      J=1
C   KNOW THAT NODE (1,1) HAS BEEN ORDERED
      IF (KRITE .EQ. 8)  GO TO 14
      I=ATRIB(5)
      J=ATRIB(6)
      IF (J .NE. 7777)  GO TO 14
      I=ATRIB(7)
      J=ATRIB(8)
   14 IF (ATRIB(1)-NSET(I,J,1)) 5,6,7
C   .LT. 0 MEANS GO TO PREDECESSOR,  .GT. 0 GO TO SUCCESSOR
    5 IA=NSET(I,J,7)
      IF (M7 .EQ. 1)  GO TO 9
      IF (IA .EQ. 9999) GO TO 9
C   SEE IF IT HAS A PREDECESSOR
      J=NSET(I,J,8)
      I=IA
      M5=1
      GO TO 14
    7 IA=NSET(I,J,5)
      IF (M5 .EQ. 1) GO TO 6
      IF (IA .EQ. 7777) GO TO 6
C   SEE IF IT HAS A SUCCESSOR
      J=NSET(I,J,6)
      I=IA
      M7=1
      GO TO 14
    6 ATRIB(5)=NSET(I,J,5)
C   (M,N) SUCCEDES (I,J) (BY CONVENTION IF THEY HAVE = TIMES)
```

```
      ATRIB(6)=NSET(I,J,6)
      ATRIB(7)=I
      ATRIB(8)=J
      NSET(I,J,5)=M
      NSET(I,J,6)=N
      AT5=ATRIB(5)
      IF (AT5 .EQ. 7777) GO TO 98
C   TEST FAILS IF (I,J) HAS A SUCCESSOR, AND THUS MUST UPDATE HIM
      AT6=ATRIB(6)
      NSET(AT5,AT6,7)=M
      NSET(AT5,AT6,8)=N
      GO TO 98
    9 ATRIB(5)=I
C   (M,N) PRECEDES (I,J)
      ATRIB(6)=J
      ATRIB(7)=NSET(I,J,7)
      ATRIB(8)=NSET(I,J,8)
      NSET(I,J,7)=M
      NSET(I,J,8)=N
      AT7=ATRIB(7)
      IF (AT7 .EQ. 9999) GO TO 98
C   TEST FAILS IF (I,J) HAS A PREDECESSOR (WHICH MUST BE UPDATED)
      AT8=ATRIB(8)
      NSET(AT7,AT8,5)=M
      NSET(AT7,AT8,6)=N
   98 DO 99 K=5,8
   99 NSET(M,N,K)=ATRIB(K)
      RETURN
      END
C
C
      SUBROUTINE EVNTS (I,J)
      COMMON/Z/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISFED, TNOW, NBDN, NBSEC(16), PCSEC(16),
     2   NRAND(128), IRAND, MRAND, KRAND
      DIMENSION ISTM(1024), IDIST(1024), IMAX(1024), ISTK(1024)
      INTEGER ATRIB, TNOW, TFIN
      II=I
      JJ=J
      JEVNT=ATRIB(2)
      I7=JEVNT
      CALL ALTER(II,JJ,I7)
      DO 1 L1=1,8
    1 ATRIB(L1)=NSET(II,JJ,L1)
      NEWBD=ATRIB(4)+1
      NSECT=ATRIB(3)
      JSUB=5*JEVNT+NEWBD
      IF (JEVNT .EQ. 0)  ATRIB(2)=1
      IF (JEVNT .EQ. 1)  ATRIB(2)=0
      KEVNT=ATRIB(2)
```

```
         KSUH=5*KFVNT+NEHHC
         ALAMD=PARAM(KSUH,JRUN)
C     ALAMD IS THE MEAN INTERARRIVAL TIME I.E. 1/LAMBDA OR 1/MU
         CALL URAND(RNUM)
         ATRIB(1)=TNOW-INT(ALAMD*ALOG(RNUM)-0.5)
C     JEVNT=0, KEVNT=1 IF NODE CHANGED FROM GOOD TO BAD, AND VICE VERSA
         ATRIB(3)=NSELT
         ATRIB(4)=NEHHD-1
         KOUNT(KSUH)=KSUNT(KSUB)+1
         KOUNT(JSUB)=KOUNT(JSUH)-1
         IF (JEVNT .EQ. 1)  GO TO 10
         NBDN=NBDN+1
C     NBDN IS  NO OF BAD NODES IN GRID
         NBSEC(NSECT)=NBSEC(NSECT)+1
C     NBSEC IS NO OF BAD NODES IN SECTION
         GO TO 12
   10    NBDN=NBDN-1
         NBSEC(NSECT)=NBSEC(NSECT)-1
   12    RSEC=NBSEC(NSECT)
         PCSEC(NSECT)=RSEC/64.0
         DO 99 I=1,2
   99    NSET(II,JJ,I)=ATRIB(I)
         IF (KRITE .EQ. 0)  GO TO 57
         IF (TNOW.LT.1800) GO TO 305
    8    LVL1=0
         DO 218 M1=1,1024
  218    ISTK(M1)=0
         DO 110 IR=1,32
         DO 110 IC=1,32
         IF (NSET(IR,IC,2).EQ.0) GO TO 110
         NX=IR+32*(IC-1)
         LVL1=LVL1+1
         ISTK(LVL1)=NX
  110    CONTINUE
         MP=LVL1+1
         DO 111 J1=MP,1024
  111    ISTK(J1)=0
         SUM=LVL1
         NLOTS=0
         NODES=0
         IF (LVL1) 33,33,30
   29    IF (LVL1) 32,32,30
   30    CALL CNVRT(ISTK(LVL1),NR,NC)
         ISTK(LVL1)=0
         LVL1=LVL1-1
         LOT=1
         LVL2=1024
         GO TO 67
  114    IF (LVL2.EQ.1024) GO TO 16
         LVL2=LVL2+1
```

```
      CALL CNVRT(ISTK(LVL2),NR,NC)
      IF (NSET(NR,NC,4).LT.4)  JKNT=JKNT+1
      ISTK(LVL2)=0
      LBT=LBT+1
   67 IF (NR-32) 81,2,2
   81 NR1=NR+1
      NC1=NC
      LINE=1
   77 NY=NR1+32*(NC1-1)
      IF (LVL1.NE.0) GO TO 117
      IF (LVL2.EQ.1024)   GO TO 16
      LBT=LBT+1024-LVL2
      LVL3=LVL2+1
      DO 302 I5=LVL3,1024
      CALL CNVRT(ISTK(I5),NR,NC)
      IF (NSET(NR,NC,4).LT.4)  JKNT=JKNT+1
  302 CONTINUE
      GO TO 16
  117 DO 20 IM=1,LVL1
      IF (ISTK(IM)-NY) 20,119,20
  119 ISTK(LVL2)=NY
      LVL2=LVL2-1
C  LVL2 IS AN OPEN SPOT
      ISTK(IM)=ISTK(LVL1)
      ISTK(LVL1)=0
      LVL1=LVL1-1
      GO TO 121
   20 CONTINUE
  121 GO TO (2,4,6,114),LINE
    2 IF (NR-1) 4,4,3
    3 NR1=NR-1
      NC1=NC
      LINE=2
      GO TO 77
    4 IF (NC-32) 5,6,6
    5 NR1=NR
      NC1=NC+1
      LINE=3
      GO TO 77
    6 IF (NC-1) 114,114,7
    7 NR1=NR
      NC1=NC-1
      LINE=4
      GO TO 77
C  HAVE PULLED OUT ALL OF BAD NBRS OF (NR,NC) AND PLACED THEM
C     AT BOTTOM OF STACK
      GO TO 114
   16 CONTINUE
      ISTM(LBT)=ISTM(LBT)+1
      NLOTS=NLOTS+1
      GO TO 29
   32 PLOTS=NLOTS
   33 CONTINUE
      IF (ISEED.GT.0) GO TO 308
      DO 311 J=1,1024
      IDIST(J)=0
  311 IMAX(J)=0
  308 CONTINUE
      ISEED=ISEED+1
      DO 301 J=1,1024
      IF (ISTM(J).EQ.0)  GO TO 301
      IDIST(J)=IDIST(J)+ISTM(J)
      MAX=J
  301 CONTINUE
      IMAX(MAX)=IMAX(MAX)+1
      IF (ISEED.LT.200)  GO TO 305
      WRITE(108,312)  TNOW
      KRUM=0
      KSUM=0
      DO 303 J=1,1024
      KSUM=KSUM+IDIST(J)
  303 KRUM=KRUM+IMAX(J)
      DO 304 J=1,1024
      IF (IDIST(J).EQ.0)  GO TO 304
      STEM=FLOAT(IDIST(J))/FLOAT(KSUM)
      WRITE(108,306) J, STEM
  304 CONTINUE
      WRITE (108,309)
      DO 307 J=1,1024
      IF (IMAX(J).EQ.0) GO TO 307
      STEM= FLOAT(IMAX(J))/FLOAT(KRUM)
      WRITE (108,310) J, STEM
  307 CONTINUE
      RETURN
  305 CONTINUE
  306 FORMAT (1H ,11HCLUMP SIZE=,I4,5X,10HFREQUENCY=,F7.5)
  310 FORMAT (1H ,11HCLUMP SIZE=,I4,5X,13HFREQ. AS MAX=,F7.5)
  309 FORMAT(1H0)
  312 FORMAT(1H0,5X,5HTIME=,I6)
   57 CALL ORDER(II,JJ)
   19 RETURN
      END
C
C
       SUBROUTINE RACK(J,K)
       COMMON/7/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NBDN, NBSEC(16), PCSEC(16),
     2   NRAND(128), IRAND, MRAND, KRAND
       INTEGER ATRIB, TNOW, TFIN
C    (J,K) IS AN INITIALLY BAD NODE
```

```
      NSET(J,K,2)=1
C   WILL UPDATE NB OF HAD NODES IN SECTOR, AND PCT HAD IN SECTOR
      NSECT=NSET(J,K,3)
      NBSEC(NSECT)=NBSEC(NSECT)+1
      RSEC=NBSEC(NSECT)
      PCSEC(NSECT)=RSEC/64.0
      DO 10 I=9,12
      II=I
      ME=NSET(J,K,II)
      IF (ME.EQ.0)  RETURN
      CALL CNVRT(ME,JROW,JCOL)
      NROW=JROW
      NCOL=JCOL
   10 NSET(NROW,NCOL,4)=NSET(NROW,NCOL,4)+1
    9 RETURN
      END
C
C
      SUBROUTINE ALTER(J,K,NCODE)
C   ALTER UPDATES NBRS OF (J,K), WHICH HAS JUST CHANGED STATE
C   THIS CORRESPONDS TO THE WORK OF  FIND  IN THE NON-RANDOM GRAPH
      COMMON/7/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NBDN, NBSEC(16), PCSEC(16)
      INTEGER ATRIB, TNOW, TFIN, ALT
      NROW=J
      NCOL=K
      DO 10 I=9,12
      II=I
      ME=NSET(NROW,NCOL,II)
      IF (ME.EQ.0)  RETURN
      CALL CNVRT(ME,JJROW,JJCOL)
      JROW=JJROW
      JCOL=JJCOL
      ATRIB(1)=NSET(JROW,JCOL,1)
      ATRIB(2)=NSET(JROW,JCOL,2)
      ATRIB(4)=NSET(JROW,JCOL,4)
      JTIME=ATRIB(1)
      JNEHB=ATRIB(4)+1
      IF (NCODE .EQ. 0) ATRIB(4)=ATRIB(4)+1
      IF (NCODE .EQ. 1) ATRIB(4)=ATRIB(4)-1
      KNEHB=ATRIB(4)+1
      JSUB=5*ATRIB(2)+JNEHB
      KSUB=5*ATRIB(2)+KNEHB
      KOUNT(KSUB)=KOUNT(KSUB)+1
      KOUNT(JSUB)=KOUNT(JSUB)-1
      ALAMD=PARAM(KSUB,JRUN)
      CALL DRAND(RNUM)
      ALT=TNOW-INT(ALAMD*ALOG(RNUM)-0.5)
    3 ATRIB(1)=ALT
      GO TO 6
    4 IF(ATRIB(1)-ALT) 5,6,7
    5 IF (NCØDE.NE.ATRIB(2)) ATRIB(1)=ALT
      GØ TØ 6
    7 IF (NCØDE.EQ.ATRIB(2)) ATRIB(1)=ALT
    6 CØNTINUE
      NSET(JRØW,JCØL,1)=ATRIB(1)
      NSET(JRØW,JCØL,4)=ATRIB(4)
      IF (JTIME.EQ.ATRIB(1))  GØ TØ 10
      CALL RMØVE(JRØW,JCØL)
      CALL ØRDER(JRØW,JCØL)
   10 CØNTINUE
    8 RETURN
      END
C
C
      SUBRØUTINE RMØVE (JRØW, JCØL)
      CØMMØN/7/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KAUNT(10), ISFED, TNØW, NBDN, NBSEC(16), PCSEC(16)
      INTEGER ATRIB, TNØW, TFIN
      DØ 10 K=1,8
   10 ATRIB(K)= NSET(JRØW,JCØL,K)
      ISUCR=ATRIB(5)
      ISUCC=ATRIB(6)
      IPRER=ATRIB(7)
      IPREC=ATRIB(8)
      IF (IPRER .EQ. 9999) GØ TØ 15
      NSET(IPRER,IPREC,5)=ISUCR
      NSET(IPRER,IPREC,6)=ISUCC
      IF (ISUCR .EQ. 7777)  GØ TØ 25
   15 NSET(ISUCR,ISUCC,7)=IPRER
      NSET(ISUCR,ISUCC,8)=IPREC
   25 RETURN
      END
C
C
      SUBRØUTINE GASE
      CØMMØN/7/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KAUNT(10), ISFED, TNØW, NBDN, NBSEC(16), PCSEC(16)
      INTEGER ATRIB, TNØW, TFIN, SUCR, SUCC
      NEXTR=1
      NEXTC=1
      JRITTE=0
      ISEED=0
    4 IF (NSET(NEXTR,NEXTC,7) .EQ. 9999)  GØ TØ 5
C   TEST ABØVE FAILS IF (NEXTR,NEXTC) IS NØT THE HEAD ØF THE LIST
      NEX=NSET(NEXTR,NEXTC,7)
      NEXTC=NSET(NEXTR,NEXTC,8)
      NEXTR=NEX
C   SEE IF HIS PREDECESSØR IS HEAD ØF LIST
      GØ TØ 4
    5 CALL RMOVE(NEXTR,NEXTC)
      ITEST=ISITON(6)
      IF(ITEST.NE.6) GO TO 598
  597 ITEST=ISITON(6)
      IF(ITEST.EQ.0)  RETURN
      GO TO 597
  598 CONTINUE
      TNOW=ATRIB(1)
      IF (TNOW .GE. TFIN(JRUN)) GO TO 17
      SUCR=ATRIB(5)
      SUCC=ATRIB(6)
      JRITE=JRITE+1
      KRITE=0
      IF (JRITE .LT. JRATE(JRUN)) GO TO 7
C   IF ABOVE TEST IS MET, WILL NOT PRINT RESULTS AT THIS EVENT TIME
      JRITE=0
      KRITE=1
C   WILL PRINT RESULTS AT THIS EVENT TIME
    7 CALL EVNTS(NEXTR,NEXTC)
      NEXTR=SUCR
      NEXTC=SUCC
      GO TO 4
   17 KRITE=1
      WRITE (108,21)
   21 FORMAT (1H0,52X,26H** FINAL REPORT FOLLOWS **)
      CALL EVNTS (NEXTR,NEXTC)
      RETURN
      END
C
C
      SUBROUTINE DATAN
      COMMON/Z/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NRON, NBSEC(16), PCSEC(16),
     2   NRAND(128), IRAND, MRAND, KRAND
      DIMENSION  MAP(64), BAD(20)
      INTEGER ATRIB, TNOW, TFIN, BAD
      REAL LAMBDA, MU
      IF ( JRUN .NE. 1)  GO TO 29
      READ (105,31) KRITE
   31 FORMAT (I1)
      IF (KRITE) 32,32,33
   32 DO 19 J=1,10
      READ (105,21)  LAMBDA, MU, N
      WRITE (108,10)  LAMBDA, MU, N
      IF (LAMBDA-MU)  6,7,8
    6 WRITE (108,9)
    9 FORMAT (50H0ERROR- LAMBDA MUST BE GREATER THAN OR EQUAL TO MU)
      JRUN=20
      RETURN
    7 P=N
```

```
      PARAM(1,J)=P/MU
      GØ TØ 11
    8 PARAM(1,J)=1.0/(LAMBDA-MU)
   11 PARAM(6,J)=1.0/MU
      DELTA=MU/5.0
      DØ 19 I=2,5
      MU=MU-DELTA
      PARAM(I,J)=1.0/(LAMBDA-MU)
   19 PARAM(I+5,J)=1.0/MU
   10 FØRMAT (4H0LAMBDA=,F7.4,5X,3HMU=,F7.4,5X,
     1    22HSIZE ØF QUEUEING RØØM=,I5)
   21 FØRMAT (2(F7.0),I5)
      GØ TØ 36
   33 DØ 34 I=1,10
   34 READ (105,35) (PARAM(J,I),J=1,10)
   35 FØRMAT(10F7.0)
   36 READ (105,22) (JRATE(I),I=1,10)
   22 FØRMAT (10I7)
      READ (105,23) (TFIN(I),I=1,10)
   23 FØRMAT (10I7)
      DØ 4 J=1,10
      WRITE (108,3)  J, JRATE(J), TFIN(J)
    4 WRITE (108,5)  (PARAM(I,J),I=1,10)
    3 FØRMAT (1H0,7HRUN NØ.,I2,10X,12HREPØRT RATE=,I7,10X,
     1  12HFINISH TIME=,I7)
    5 FØRMAT (1H ,11HPARAMETERS=,10(F7.2,5X))
   29 READ (105,25) NIBN
      KRITE=8
   25 FØRMAT (I4)
C   NIBN IS NØ ØF INITIALLY BAD NØDES
      NBLN=NIBN
      READ (105,38) ISEED
   38 FØRMAT(I1)
      WRITE (108,26) JRUN, NIBN, ISEED, JRATE(JRUN), TFIN(JRUN),
     1                 LRAND,MRAND,KRAND,NRAND(1)
   26 FØRMAT (1H1,22X,13HRANDØM GRAPH ,
     1                    7HRUN NØ.,I2,10X,27HNØ. ØF INITIALLY BAD NØDES=,
     2  I4,10X,12HFIND ØPTIØN=,I1/1H0,4X,12HREPØRT RATE=,I7,10X,
     3  12HFINISH TIME=,I7,10X,13HRANDØM SEEDS=,4(I9,1X))
      WRITE (108,30) (PARAM(I,JRUN),I=1,10)
   30 FØRMAT (1H0,9HPARAM(1)=,F7.2,10X,9HPARAM(2)=,F7.2,10X,9HPARAM(3)=,
     1  F7.2,10X,9HPARAM(4)=,F7.2,10X,9HPARAM(5)=,F7.2/1H0,9HPARAM(6)=,
     2  F7.2,10X,9HPARAM(7)=,F7.2,10X,9HPARAM(8)=,F7.2,10X,9HPARAM(9)=,
     3  F7.2,10X,10HPARAM(10)=,F7.2)
C   WILL NØW SET SECTIØN NØS AND RESET ØTHER ELEMENTS ØF NSET
      LN=0
      DØ 14 ML=1,25,8
      ML7=ML+7
      DØ 14 I=1,25,8
      I7=I+7
```

```
          LN=LN+1
C     LN WILL BE THE SECTION NØ
          DØ 14 N=ML,ML7
          DØ 14 J=1,17
          DØ 13 K=1,2
   13     NSET(N,J,K)=0
          NSET(N,J,3)=LN
          DØ 14 M=4,8
   14     NSET(N,J,M)=0
          DØ 37 I=1,10
   37     KØUNT(I)=0
          DØ 17 I=1,16
          NØSEC(I)=0
   17     PCSEC(I)=0
          READ (105,47) IDATA
   47     FØRMAT (I1)
          IF (IDATA .EQ. 1) GØ TØ 99
C     IDATA=1 IF USING ALTERNATE FØRM ØF INPUT ØF INITIALLY BAD NØDES
          IF (NBDN .EQ. 0)  GØ TØ 44
          WRITE (108,53)
   53     FØRMAT (1H0,19X,33HLIST ØF INITIAL BAD NØDES FØLLØWS)
          DØ 43 I=1,NBDN,10
          READ (105,45) (BAD(J),J=1,20)
   45     FØRMAT (20(I2))
          WRITE (108,54)  (BAD(J),J=1,20)
   54     FØRMAT(1H0,10(2H (,I2,1H,,I2,2H) ))
          DØ 43 M=1,10
          J=BAD(2*M-1)
          K=BAD(2*M)
C     BAD NØDE IS (J,K)
          IF (J .EQ. 44) GØ TØ 44
C     J=44 MARKS END ØF LIST ØF INITIAL BAD NØDES
          CALL HACK (J,K)
   43     CØNTINUE
   44     DØ 77 M=1,32
          DØ 77 N=1,32
          JEVNT=NSET(M,N,2)
          NEHBD=NSET(M,N,4)
          JSUB=5*JEVNT+NEHBD+1
          ALAMD=PARAM(JSUB,JRUN)
          CALL DRAND(RNUM)
          NSET(M,N,1)=-INT(ALAMD*ALØG(RNUM)-0.5)
   77     KØUNT(JSUB)=KØUNT(JSUB)+1
          WRITE (108,64) (KØUNT(I),I=1,10)
   64     FØRMAT (1H0,7HKNT(1)=,I4,2X,7HKNT(2)=,I4,2X,7HKNT(3)=,I4,2X,
         1   7HKNT(4)=,I4,2X,7HKNT(5)=,I4,2X,7HKNT(6)=,I4,2X,7HKNT(7)=,I4,
         2   2X,7HKNT(8)=,I4,2X,7HKNT(9)=,I4,2X,8HKNT(10)=,I4,2X)
          IF (NSET(1,1,1)-NSET(1,2,1)) 78,78,79
   78     NSET(1,1,5)=1
          NSET(1,1,6)=2
```

```
      NSET(1,1,7)=9999
      NSET(1,1,8)=9999
      NSET(1,2,5)=7777
      NSET(1,2,6)=7777
      NSET(1,2,7)=1
      NSET(1,2,8)=1
      GØ TØ 83
   79 NSET(1,1,5)=7777
      NSET(1,1,6)=7777
      NSET(1,1,7)=1
      NSET(1,1,8)=2
      NSET(1,2,5)=1
      NSET(1,2,6)=1
      NSET(1,2,7)=9999
      NSET(1,2,8)=9999
   83 ML=0
      DØ 80 I=1,32
      II=I
      DØ 80 J=1,32
      JJ=J
      ML=ML+1
      IF (ML .LE. 2) GØ TØ 80
      ATRIB(1)=NSET(II,JJ,1)
      CALL ØRDER(II,JJ)
   80 CØNTINUE
      KRITE=0
      RETURN
   99 WRITE (108,101)
  101 FØRMAT(1H0/1H0,16X,33HINITIAL GRID (8'S MARK BAD NØDES)/1H0/1H0)
      DØ 127 I=1,32,2
      READ (105,121) (MAP(J),J=1,64)
  121 FØRMAT (64I1)
C     TWØ RØWS ØF THE GRID CØMPRISE ØNE LINE ØF DATA
      WRITE (108,122) (MAP(J),J=1,32)
  122 FØRMAT (1H ,32(I1,1X))
C     BAD NØDES = 8, GØØD NØDES = 1
      WRITE(108,122) (MAP(J),J=33,64)
      DØ 123 M=1,32
      IF (MAP(M) .EQ. 1) GØ TØ 123
      J=I
      K=M
      CALL BACK (J,K)
  123 CØNTINUE
      DØ 127 M=33,64
      IF (MAP(M) .EQ. 1) GØ TØ 127
      J=I+1
      K=M-32
      CALL BACK (J,K)
  127 CØNTINUE
      GØ TØ 44
```

```
C    JOINS PROGRAM WHERE REGULAR FORM OF INPUT ENDED
      END
C
C
      SUBROUTINE XBRAN (N,K,NDS4)
      COMMON/7/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISFED, TNOW, NBON, NBSEC(16), PCSEC(16)
C    N IS HUNGUP, K HAS 4 NBRS NONE OF WHICH IS N
      NDS4=0
      NN=N
      CALL CNVRT(NN,NR,NC)
    3 KK=K
      CALL CNVRT(KK,KR,KC)
C    HANGUP HAS OCCURRED BECAUSE ONLY AVAILABLE NODES ARE ALREADY
C     NBRS OF (NR,NC), OR ELSE THERE ARE NO AVAILABLE NODES
      WRITE(10R,2) N, K
    2 FORMAT(1H0,2HI=,I4,2HJ=,I4)
      DO 10 IP=9,12
      L=NSET(NR,NC,IP)
      IF (L .EQ. 0) GO TO 10
      CALL CNVRT(L,LR,LC)
      IF (NSET(LR,LC,3) .LT. 4) GO TO 14
   10 CONTINUE
C    ALL OF HIS NBRS ARE FULL
      GO TO 75
C    L IS A NBR OF N WHO HAS FEWER THAN 4 NBRS (IF AT STATEMENT 14)
   14 KX=12
   19 I=NSET(KR,KC,KX)
      IF (I .EQ. 0) GO TO 27
      IF (I.EQ.L)  GO TO 27
      CALL CNVRT (I,IR,IC)
      DO 20 KG=9,12
      IF (NSET(IR,IC,KG) .EQ. L) GO TO 27
   20 CONTINUE
      GO TO 25
   27 KX=KX-1
      IF (KX .GE. 9) GO TO 19
   46 IF (K-(N-1)) 47,48,48
   47 K=K+1
      GO TO 49
   48 K=K-1
   49 CALL CNVRT(K,KR,KC)
      DO 30 KM=9,12
      IF (NSET(KR,KC,KM) .EQ. N)  GO TO 46
   30 CONTINUE
      GO TO 3
C    CAN EXCHANGE BRANCHES IF AT STATEMENT 25
   25 IF (NSET(NR,NC,3) .EQ. 3) NDS4=NDS4+1
      IF (NSET(LR,LC,3) .EQ. 3) NDS4=NDS4+1
      CALL REORD(KR,KC,IR,IC)
      NSET(KR,KC,3)=NSET(KR,KC,3)-1
      NSET(IR,IC,3)=NSET(IR,IC,3)-1
      CALL PLACE(NR,NC,KR,KC)
      CALL PLACE(IR,IC,LR,LC)
      RETURN
   75 KX=12
   76 I=NSET(KR,KC,KX)
      WRITE(108,4)
    4 FORMAT(5HOST76)
      IF (I .EQ. 0) GO TO 78
      CALL CNVRT(I,IR,IC)
      DO 77 KG=9,12
      IF (NSET(IR,IC,KG) .EQ. NN) GO TO 78
   77 CONTINUE
      CALL REORD(KR,KC,IR,IC)
      NSET(KR,KC,3)=NSET(KR,KC,3)+1
      NSET(IR,IC,3)=NSET(IR,IC,3)+1
      CALL PLACE(NR,NC,KR,KC)
      CALL PLACE (NR,NC,IR,IC)
      IF (NSET(NR,NC,3) .EQ. 4)  NDS4=NUS4+1
      GO TO 79
   78 KX=KX-1
      IF (KX .GE. 9) GO TO 76
      GO TO 46
   79 RETURN
      END
C
C
      SUBROUTINE REORD(NR,NC,KR,KC)
      COMMON/Z/NSET(32,32,12)
      WRITE(108,1)
    1 FORMAT(13HOREORD CALLED)
      WRITE(108,5) NR, NC
    5 FORMAT(6HONODE=,I2,2X,I2)
      WRITE(108,6)  (NSET(NR,NC,I),I=3,12)
    6 FORMAT(11HOATHIR3-12=,10(3X,I4))
      WRITE(108,5) KR, KC
      WRITE(108,6)  (NSET(KR,KC,I),I=3,12)
      N=NR+32*(NC-1)
      K=KR+32*(KC-1)
C  REORD PUTS N AND K AT THE END OF THE OTHER'S LIST OF NBRS
   27 NL=NSET(NR,NC,3)
      GO TO (8,2,3,4), NL
    2 IF (NSET(NR,NC,10) .EQ. K) GO TO 8
      NSET(NR,NC,9)=NSET(NR,NC,10)
      GO TO 8
    3 IF (NSET(NR,NC,11) .EQ. K) GO TO 8
      IF (NSET(NR,NC,10)-K) 9,10,9
   10 NSET(NR,NC,10)=NSET(NR,NC,11)
      GO TO 8
    9 NSET(NR,NC,9)=NSET(NR,NC,11)
      GO TO 8
    4 IF (NSET(NR,NC,12) .EQ. K) GO TO 8
      IF (NSET(NR,NC,11)-K) 12,13,12
   13 NSET(NR,NC,11)=NSET(NR,NC,12)
      GO TO 8
   12 IF (NSET(NR,NC,10)-K) 15,16,15
   16 NSET(NR,NC,10)=NSET(NR,NC,12)
      GO TO 8
   15 NSET(NR,NC,9)=NSET(NR,NC,12)
    8 IF (K .EQ. N) GO TO 19
      K=N
      MR=NR
      MC=NC
      NR=KR
      NC=KC
      GO TO 27
   19 NR=MR
      NC=MC
      RETURN
      END
C
C
      SUBROUTINE CNVRT(N,IROW,ICOL)
      DO 10 I=1,32
      IF (N-32)  5,5,10
   10 N=N-32
    5 IROW=N
      ICOL=I
      RETURN
      END
C
C
      SUBROUTINE PLACE(NR,NC,KR,KC)
      COMMON/7/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NBDN, NOSEC(16), PCSEC(16)
      LR=NR
      LC=NC
      IR=KR
      IC=KC
      IF ((LR.EQ.IR) .AND. (LC.EQ.IC))  GO TO 24
      IF ((NSET(LR,LC,3).GE.4).OR.(NSET(IR,IC,3).GE.4)) GO TO 24
   27 LEVEL=NSET(LR,LC,3)+1
C   LEVEL IS NO OF BRANCHES + 1
      K=IR+32*(IC-1)
      GO TO (1,2,3,4),LEVEL
    1 NSET(LR,LC,9)=K
      GO TO 19
    2 NSET(LR,LC,10)=K
      GO TO 19
```

```
  3   NSET(LR,LC,11)=K
      GO TO 19
  4   NSET(LR,LC,12)=K
 19   NSET(LR,LC,3)=LEVEL
      IF (LR.EQ.KR.AND.LC.EQ.KC) GO TO 24
      LR=KR
      LC=KC
      IR=NR
      IC=NC
      GO TO 27
 24   RETURN
      END
C
C
      SUBROUTINE GRAPH
      COMMON/Z/NSET(32,32,12),PARAM(10,10),JRATE(10),TFIN(10),ATRIB(8),
     1 KRITE, JRUN, KOUNT(10), ISEED, TNOW, NBDN, NBSEC(16), PCSEC(16)
      NDS4=0
C   NDS4=NO OF NODES THAT HAVE 4 NBRS
      DO 1 I=1,32
      DO 1 J=1,32
      NSET(I,J,3)=0
      DO 1 K=9,12
  1   NSET(I,J,K)=0
      DO 15 NC=1,32
      DO 15 NR=1,32
C   NC IS COL NO, NR IS ROW NO
      N=NR+32*(NC-1)
      IF (NSET(NR,NC,3)-3) 5,5,15
C   NSET(NR,NC,3) TELLS HOW MANY NBRS NODE (NR,NC) HAS AT THIS TIME
C   AT END OF SUBROUTINE NSET(I,J,3)=4 FOR EVERY NODE (I,J)
  5   NREP=4-NSET(NR,NC,3)
      DO 12 KG=1,NREP
  6   CALL DRAND(RNUM)
      NUM=INT((FLOAT(1023-NDS4))*RNUM+0.5)
C   PICK UP NUM-TH AVAILABLE NODE IN NSET AND BEGIN SEARCH
      MA=0
 27   DO 14 KC=NC,32
      DO 14 KR=1,32
      IF((KC.EQ.NC  ) .AND. (KR.EQ.NR  ))  GO TO 14
C   ABOVE TEST PREVENTS IT FROM PLACING BRANCH ON ITSELF
      IF (NSET(KR,KC,3)-3)  7,7,14
  7   MA=MA+1
      IF (MA-NUM) 14,9,9
  9   DO 13  LN=9,12
      IF (NSET(KR,KC,LN).EQ.N) GO TO 14
C   TEST IS MET IF THERE IS ALREADY A BRANCH CONNECTING THEM
 13   CONTINUE
      NNR=NR
      NNC=NC
      KKR=KR
      KKC=KC
      CALL PLACE(NNR,NNC,KKR,KKC)
C   PLACE UPDATES NSET WITH THE NEW BRANCH
      IF (NSET(NNR,NNC,3).EQ.4) NDS4=NDS4+1
      IF (NSET(KKR,KKC,3).EQ.4) NDS4=NDS4+1
      GO TO 12
   14 CONTINUE
      IF (MA .GT. 1500) GO TO 100
      MA=1600
      GO TO 27
C   ABOVE, FAILED TO ADD A BRANCH, MUST TRY AGAIN AMONG FIRST NUM
C    AVAILABLE NODES
   12  CONTINUE
   43 CONTINUE
      GO TO 15
C   MUST PERFORM BRANCH EXCHANGE IF AT STATEMENT 100
  100 KRITE=NSET(NR,NC,3)
      IF (NSET(NR,NC,3) .EQ. 4) GO TO 15
      CALL DRAND(RNUM)
      J=INT(FLOAT(N)*RNUM)
      IF (J .EQ. 0) GO TO 100
      IF (J .EQ. N) GO TO 100
      DO 101 K=9,12
      IF (NSET(NR,NC,K) .EQ. J) GO TO 100
C   I.E., IF THEY ARE ALREADY NBRS, TRY AGAIN
  101 CONTINUE
C   ANY HANGUPS INVOLVE 4 OR FEWER NODES (2 IS MOST LIKELY NO.)
      WRITE(108,2) N, J
    2 FORMAT(1H0,2HN=,I4,2HJ=,I4)
      CALL XBRAN (N,J,INCR)
      NDS4=NDS4+INCR
      IF (NSET(NR,NC,3) .EQ. 4) GO TO 15
      IF (KRITE-NSET(NR,NC,3))   100,16,16
   16 WRITE(108,35)
   35 FORMAT(8H0NO HOPE)
   15 CONTINUE
      WRITE(108,37) NDS4
   37 FORMAT (1H ,26HNO OF NODES WITH DEGREE 4=,I4)
    3 DO 8 I=1,32
      DO 8 J=1,32
      IF (NSET(I,J,3).NE.4) WRITE(108,10) I,J,NSET(I,J,3)
      DO 8 K=9,12
      IF(NSET(I,J,K).EQ.0)  WRITE(108,10) I,J,K
    8 CONTINUE
   10 FORMAT(7H0NODE (,I2,1H,,I2,11H) HAS ONLY ,I1,5H NBRS)
    4 RETURN
      END
 SYMBOL
          DEF       ISITON
```

```
         ISITON    CAL2,1    C
                   RD,0      C
                   STCF      3
                   CAL2,1    1
                   SCS,3     4
                   LW,4      13
                   LW,13     1,4
                   AND,3     *13
                   B         2,4
                   END
```

Message Transfer Network

```
C    MAIN PROGRAM
      COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
     1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
     2   MRAND, KRAND, NOW, JRATE, ND(64,51,3), MESG(64,64)
      COMMON IG1, IG2, IG3, NSIDE, NSQ, ISEED
      RFAD (105,8) LRAND, MRAND, KRAND, NRAND(1)
      DO 20 I=2,128
   20 NRAND(I)=NRAND(I-1)+2*I
    1 RFAD (105,2) SIGMA, RMU, SIGR, N, (MULT(K),K=1,6), JRATE, NSIDE,
     1     ISEFD
    2 FORMAT (3F7.0,7I2,I5,I2,I1)
      NSQ=NSIDE*NSIDE
      DO 3 I=1,64
      NS(I)=N
      LSM(I)=0
      NXND(I)=0
      DO 4 J=1,64
    4 MESG(I,J)=0
      DO 5 J=1,5
    5 LS(I,J)=0
      NEX(I,1)=0
      NEX(I,2)=0
      DO 6 J=1,51
      DO 6 K=1,3
    6 ND(I,J,K)=0
    3 CONTINUE
      NXND(65)=0
      DO 7 I=1,4
    7 MESS(I)=0
      NEX(65,1)=0
      NEX(65,2)=0
      NOW=0
    8 FORMAT (4I9)
      WRITE (108,10) SIGMA, RMU, SIGR, N, (MULT(K),K=1,6), JRATE,LRAND,
     1   NSQ, ISEED
      SIGMA=1./SIGMA
      SIGR=1./SIGR
      RMU=1./RMU
      CALL GASP
      GO TO 1
   10 FORMAT (1H1,   6HSIGMA=,F7.5,3X,3HMU=,F7.5,3X,6HSIGRE=,F7.5,3X,
     1   2HN=,I2,3X,8HMULT(I)=,6(I2,2X),5HRATE=,I5,3X,5HRAND=,I9,
     2     3X,4HNSQ=,I2,3X,5HPREF=,I1)
      STOP
      END
C
C
      SUBROUTINE DRAND(RNUM)
      COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
     1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
     2   MRAND, KRAND, NOW, JRATE, ND(64,51,3), MESG(64,64)
      COMMON IG1, IG2, IG3, NSIDE, NSQ
      LRAND=LRAND*65539
      MRAND=MRAND*33554433
      J=1+IABS(LRAND)/16777216
      RNUM=.5+FLOAT(NRAND(J)+LRAND+MRAND)*.23283064E-9
      KRAND=KRAND*362436069
      NRAND(J)=KRAND
      RETURN
      END
C
C
      SUBROUTINE CNVRT(J,IROW,ICOL)
      COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
     1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
     2   MRAND, KRAND, NOW, JRATE, ND(64,51,3), MESG(64,64)
      COMMON IG1, IG2, IG3, NSIDE, NSQ
      M=J
      DO 10 I=1,NSIDE
      IF (J-NSIDE)  5,5,10
   10 J=J-NSIDE
    5 ICOL=J
      IROW=I
      J=M
      RETURN
      END
C
C
      SUBROUTINE ROUTE (I,IDEST,IC)
      COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
     1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
     2   MRAND, KRAND, NOW, JRATE, ND(64,51,3), MESG(64,64)
      COMMON IG1, IG2, IG3, NSIDE, NSQ
      IG1=0
      CALL CNVRT(I,IROW,ICOL)
      CALL CNVRT(IDEST,JROW,JCOL)
      IF (JROW-IROW) 1,3,2
    1 IF (IROW-JROW-NSIDE/2) 4,4,5
    4 IC=I-NSIDE
      GO TO 6
    5 IC=I+NSIDE
      GO TO 6
    2 IF (JROW-IROW-NSIDE/2) 5,5,4
    3 IF (JCOL-ICOL) 7,8,9
    7 IF (ICOL-JCOL-NSIDE/2) 10,10,11
   10 IC=I-1
      GO TO 6
   11 IC=I+1
      GO TO 6
    9 IF (JCOL-ICOL-NSIDE/2) 11,11,10
```

```
      8   WRITE (108,12) I, IDFST
     12   FORMAT (1H1,10X,11HERROR-ROUTE,2(3X,I2))
          RETURN
      6   IF (IC.GT.NSG) IC=IC-NSQ
          IF (IC.LT.1) IC=IC+NRQ
          IF (IG1) 15,16,15
     16   IG1=IC
          IF (JROW.EQ.IROW) RETURN
          IF (JCOL-ICOL) 3,18,3
     15   CALL DRAND (RNUM)
          IF (RNUM.GT.0.5) RFTURN
          K=IC
          IC=IG1
          IG1=K
     18   RETURN
          END
C
C
          SUBROUTINE INVERT
          COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
         1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
         2   MRAND, KRAND, NAW, JRATE, ND(64,51,3), MESG(64,64)
          COMMON IG1, IG2, IG3, NSIDE, NSQ, ISEED
          I=IG1
          K=IG2
          IF (K.EQ.1) IC=I+1
          IF (K.EQ.2) IC=I-1
          IF (K.EQ.3) IC=I+NSIDE
          IF (K.EQ.4) IC=I-NSIDE
          IF (IC.GT.NSQ) IC=IC-NSQ
          IF (IC.LT.1) IC=IC+NSQ
          IG1=IC
          RETURN
          END
C
C
          SUBROUTINE GASP
          COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
         1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
         2   MRAND, KRAND, NAW, JRATE, ND(64,51,3), MESG(64,64)
          COMMON IG1, IG2, IG3, NSIDE, NSQ
          INEXT=0
          DO 5 I=1,NSQ
          CALL DRAND(RNUM)
          ND(I,51,1)=-INT(SIGMA*ALOG(RNUM))+1
          NEX(I,1)=ND(I,51,1)
          NEX(I,2)=51
          IG1=I
          CALL ORDERN
      5   CONTINUE
      JTIME=0
 1    CALL EVNTS
 4    JTIME=JTIME+1
      IF (JTIME.LT.JRATE) GO TO 10
 3    NBDN=0
      DO 30 I=1,NSQ
      IF (NS(I).EQ.0) NBDN=NBON+1
30    CONTINUE
      PCT=FLOAT(NBDN)/FLOAT(NSQ)
      WRITE (108,14) NAW, NBDN, PCT
14    FORMAT (1H0///4X,5HTIME=,I6,3X,5HNBDN=,I2,3X,4HPCT=,F5.3//)
      IF (NSIDE.EQ.2) WRITE(108,32) (NS(I),I=1,4)
      IF (NSIDE.EQ.3) WRITE(108,33) (NS(I),I=1,9)
      IF (NSIDE.EQ.4) WRITE(108,34) (NS(I),I=1,16)
      IF (NSIDE.EQ.5) WRITE(108,35) (NS(I),I=1,25)
      IF (NSIDE.EQ.6) WRITE(108,36) (NS(I),I=1,36)
      IF (NSIDE.EQ.7) WRITE(108,37) (NS(I),I=1,49)
      IF (NSIDE.EQ.8) WRITE(108,38) (NS(I),I=1,64)
32    FORMAT (2(2(4X,I2)/))
33    FORMAT (3(3(4X,I2)/))
34    FORMAT (4(4(4X,I2)/))
35    FORMAT (5(5(4X,I2)/))
36    FORMAT (6(6(4X,I2)/))
37    FORMAT (7(7(4X,I2)/))
38    FORMAT (8(8(4X,I2)/))
      JTIME=0
10    CONTINUE
      ITEST=ISITON(1)
      IF (ITEST.NE.1) GO TO 22
23    ITEST=ISITON(1)
      IF (ITEST.NE.0) GO TO 23
      IF (MULT(1)-1) 24,25,24
24    DO 26 I=1,5
26    MULT(I)=1
      GO TO 28
25    DO 27 I=1,5
27    MULT(I)=10
28    WRITE (108,29) MULT(1)
29    FORMAT (1H0,15X,10HMULT(1-5)=,I4)
22    CONTINUE
      ITEST=ISITON(4)
      IF (ITEST.NE.4) GO TO 803
      READ (101,6) I
 6    FORMAT (I2)
      WRITE (108,7) I
 7    FORMAT (1H0,17HCONTENTS OF NODE ,I2)
      DO 800 I1=1,51
      WRITE (108,801) I1, ND(I,I1,1)
      NZ=ND(I,I1,2)
S         LH,4      NZ
```

```
S          STW,4     ILENTH
S          LI,1      2
S          LB,4      NZ,1
S          STW,4     IDEST
S          LI,1      3
S          LB,4      NZ,1
S          STW,4     IC
        WRITE (108,802) ILENTH, IDEST, IC
        NZ=ND(I,I1,3)
S          LH,4      NZ
S          STW,4     ISUCC
S          LI,1      1
S          LH,4      NZ,1
S          STW,4     IQUEUE
        WRITE (108,802) ISUCC, IQUEUE
    800 CONTINUE
    801 FORMAT (1H0,2X,I2,3X,I9)
    802 FORMAT (1H ,7X,3I9)
    803 CONTINUE
        ITEST=ISITON(2)
        IF (ITEST.NE.2) GO TO 11
     12 ITEST=ISITON(2)
        IF (ITEST.NE.0)  GO TO 12
        RETURN
     11 CONTINUE
        ITEST=ISITON(8)
        IF (ITEST.NE.8) GO TO 1
      2 ITEST=ISITON(8)
        IF (ITEST.NE.0)  GO TO 2
        WRITE (108,20)
        WRITE (108,21) ((MERG(I,J),J=1,64),I=1,64)
        RETURN
     20 FORMAT (1H1,56X,22HMESS GEN (ORIGIN,DEST))
     18 FORMAT (1H1,56X,22HLINE BLOCKING DISCARDS)
     21 FORMAT (    4(32(1X,I3)/)//)
     16 FORMAT (    8(16(3X,I5)/)//)
     15 FORMAT (1H1,56X,23HNODAL BLOCKING DISCARDS)
        END
C
C
        SUBROUTINE EVNTR
        COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
       1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
       2   MRAND, KRAND, NOW, JRATE, ND(64,51,3), MERG(64,64)
        COMMON IG1, IG2, IG3, NSIDE, NRQ
        I=INEXT
        NOW=NEX(I,1)
        INEXT=NXND(I)
        NXND(I)=0
        J=NEX(I,2)
```

```
C   THIS IS MESSAGE NO IN QUEUE(I) WHICH HAS NEXT COMPLETION TIME
      IF (J.EQ.0) WRITE (108,999)
  999 FORMAT (1H1, 7HERREVNT)
C   TIE UP LIST
      NZ=ND(I,J,3)
S         LH,4       NZ
S         STW,4      ISUCC
S         LI,1       1
S         LH,4       NZ,1
S         STW,4      IQUEUE
      NEX(I,2)=ISUCC
      ND(I,J,3)=IQUEUE
      IF (ISUCC.NE.0)   GO TO 5
      NEX(I,1)=0
      GO TO 1
    5 NEX(I,1)=ND(I,ISUCC,1)
    1 CONTINUE
      IF (J.NE.51) GO TO 12
      IG1=I
      CALL MESSAG
      IG2=51
      CALL ORDERQ
      IF ((NS(I).EQ.0).OR.(LSM(I).NE.0)) GO TO 300
      GO TO 865
   12 CONTINUE
      NZ=ND(I,J,2)
S         LH,4       NZ
S         STW,4      ILENTH
S         LI,1       2
S         LB,4       NZ,1
S         STW,4      IDEST
S         LI,1       3
S         LB,4       NZ,1
S         STW,4      IC
C   IC IS NEXT NODE TO WHICH MESSAGE MUST BE SENT
C   IDEST' IS ITS FINAL DESTINATION
      ICS=IC
      IF (ILENTH.EQ.0) GO TO 400
C   ILENTH=0 MEANS DEPARTURE
      IF (LSM(I).EQ.J)  LSM(I)=0
      IF (IDEST.EQ.I) GO TO 105
C   NOW WILL SEE IF THIS IS A RETRY
      DO 2 K=1,4
      IF (LS(I,K).EQ.-J)  GO TO 3
    2 CONTINUE
      GO TO 4
    3 IF (NS(IC).EQ.0) GO TO 150
C   NS(IC)=0 IF NODE IC IS BLOCKED
      GO TO 200
    4 CALL ROUTE(I,IDEST,IC)
```

```
      KCH=0
    6 ND(I,J,2)=ND(I,J,2)-ICS+IC
      JZ=IC-I
      IF ((JZ.EQ. 1).OR.(JZ.EQ.(-NSQ+1))) K=1
      IF ((JZ.EQ.-1).OR.(JZ.EQ.( NSQ-1))) K=2
      IF ((JZ.EQ. NSIDE).OR.(JZ.EQ.(-NSQ+NSIDE))) K=3
      IF ((JZ.EQ.-NSIDE).OR.(JZ.EQ.( NSQ-NSIDE))) K=4
      IF (K.LT.1.OR.K.GT.4)  GO TO 197
      ND(I,J,3)=0
      GO TO 199
  197 WRITE (108,198)  K
  198 FORMAT (1H1,7HERROR-X,3X,2HK=,I5)
      RETURN
  199 CONTINUE
      IF (NS(IC).NE.0)  GO TO 7
      IF (KCH.EQ.1)  GO TO 8
      IF (KCH.EQ.2)  GO TO 7
      KCH=1
      ICS=IC
      IC=IG1
      GO TO 6
    8 K=I+1
      IF (K.GT.NSQ) K=K-NSQ
      IF (NS(K).EQ.0)  GO TO 9
   17 ICS=IC
      IC=K
      GO TO 6
    9 K=I-1
      IF (K.LT.1)  K=K+NSQ
      IF (NS(K).EQ.0)  GO TO 10
      GO TO 17
   10 K=I+NSIDE
      IF (K.GT.NSQ) K=K-NSQ
      IF (NS(K).EQ.0)  GO TO 11
      GO TO 17
   11 K=I-NSIDE
      IF (K.LT.1)  K=K+NSQ
      IF (NS(K).NE.0) GO TO 17
      KCH=2
      GO TO 17
    7 CONTINUE
      IF (LS(I,K).NE.0) GO TO 100
      IF (NS(IC).NE.0) GO TO 200
      LS (I,K)=-J
      ND(I,J,3)=0
      GO TO 150
  200 LS (I,K)=J
   15 ITIME=NOW+ILENTH*MULT(K)
   14 ND(I,J,1)=ITIME
  210 IF (K.EQ.5) GO TO 250
```

```
          DO 211 JZ=1,N
          IF (ND(IC,JZ,2).EQ.0) GO TO 212
      211 CONTINUE
          WRITE (108,213)
      213 FORMAT (1H1,14HERROR IN EVNTS)
          RETURN
C     JZ IS THE NUMBER OF A FREE SPACE IN THE QUEUE
      212 NS (IC)=NS(IC)-1
          ND(IC,JZ,1)=ITIME+1
          ND(IC,JZ,2)=ND(I,J,2)-IC
          ND (IC,JZ,3)=0
          MSTOR=NEX(IC,1)
          IG1=IC
          IG2=JZ
          CALL ORDERQ
          IF (MSTOR.EQ.NEX(IC,1)) GO TO 215
C     REMOVE IC FROM LINKED LIST AMONG NODES
          IF (INEXT.EQ.0) GO TO 219
          IF (IC.EQ.INEXT)  GO TO 219
          MSTOR=INEXT
          DO 214 M1=1,65
          IF (MSTOR.EQ.0) GO TO 218
          IF (MSTOR.EQ.IC) GO TO 217
          MSTOR1=MSTOR
      214 MSTOR=NXND(MSTOR)
          GO TO 218
      219 INEXT=NXND(IC)
          NXND(IC)=0
          GO TO 218
      217 NXND(MSTOR1)=NXND(IC)
          NXND(IC)=0
      218 IG1=IC
          CALL ORDERN
      215 CONTINUE
      250 NZ=ND(I,J,2)
S             LI,1        1
S             LH,4        NZ,1
S             STW,4       NZ
          ND(I,J,2)=NZ
          NZ=ND(I,J,3)
S             LI,1        1
S             LH,4        NZ,1
S             STW,4       IQUEUE
          ND(I,J,3)=IQUEUE
          IG1=I
          IG2=J
          CALL ORDERQ
      300 IG1=I
          CALL ORDERN
          RETURN
```

```
  865 CONTINUE
      IDEST=MESS(2)
      ILENTH=MESS(3)
      MESG(I,IDEST)=MESG(I,IDEST)+1
C   2**16=65,536
      ITIME=NOW+ILENTH*MULT(6)
      DO 68  JZ=1,N
      IF (ND(I,JZ,2).EQ.0) GO TO 70
   68 CONTINUE
      WRITE (108,71) I
   71 FORMAT (1H1,7HERROR=A,2X,I3)
      RETURN
   70 ND(I,JZ,1)=ITIME
S         LW,5        ILENTH
S         STH,5       4
S         LW,5        IDEST
S         LI,1        2
S         STB,5       4,1
S         STW,4       NZ
      ND(I,JZ,2)=NZ
      ND (I,JZ,3)=0
      NS(I)=NS(I)-1
      LSM(I)=JZ
      IG1=I
      IG2=JZ
      CALL ORDERQ
      GO TO 300
  105 K=5
      ND(I,J,3)=0
      IF (LS(I,K).EQ.0) GO TO 200
  100 LSREP=LS(I,K)
      IF (LS(I,K).LT.0) LSREP=-LS(I,K)
      KC=0
  106 NZ=ND(I,LSREP,3)
      KC=KC+1
      IF (KC.GT.(N/4+1).AND.K.NE.5) GO TO 110
S         LI,1        1
S         LH,4        NZ,1
S         STW,4       IQUEUE
      IF (IQUEUE.EQ.0) GO TO 107
      LSREP=IQUEUE
      GO TO 106
  107 ND(I,LSREP,3)=ND(I,LSREP,3)+J
      ND (I,J,3)=0
  150 ND(I,J,1)=0
      GO TO 300
  110 K=K+1
      IF (K.GT.4) K=1
      IG1=I
      IG2=K
      CALL INVERT
      ICS=IC
      IC=IG1
      NO(I,J,2)=NO(I,J,2)-ICS+IC
      GO TO 7
  400 NWAIT=IQUEUE
      DO 401 K=1,5
      IF (LS(I,K).EQ.J) GO TO 403
  401 CONTINUE
      WRITE (108,402)
  402 FORMAT (1H1,6HERROR2)
      RETURN
  403 NO(I,J,1)=0
      NO(I,J,2)=0
      NO(I,J,3)=0
      LS (I,K)=0
      NS(I)=NS(I)+1
      IF (NS(I).NE.1) GO TO 440
      IG1=I
      CALL RETRY
  440 IF (NWAIT.NE.0) GO TO 450
      GO TO 300
  450 J=NWAIT
      IF (K.EQ.5) GO TO 460
      IF (NS(IC).GT.0) GO TO 460
      LS(I,K)=-J
      GO TO 300
  460 NZ=NO(I,J,2)
S         LH,4        NZ
S         STH,4       ILENTH
      GO TO 200
      END
C
C
      SUBROUTINE RETRY
      COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
     1   NEX(65,2), NXNO(65), MESR(4), LSM(64), NRANO(128), LRANO,
     2   MRAND, KRAND, NRW, JRATE, NO(64,51,3), MESG(64,64)
      COMMON IG1, IG2, IG3, NSIDE, NSQ, ISEED
      DIMENSION J1(4), K1(4), JF(4), KF(4)
      I=IG1
      DO 10 L=1,4
      JF(L)=0
      KF(L)=0
      J1(L)=0
   10 K1(L)=0
      J=I-1
      IF (J.EQ.0) J=NSQ
      K=1
      M=0
      MF=0
      IF (LS(J,K).GE.0) GO TO 1
      M=1
      J1(M)=J
      K1(M)=K
      IS=1
      GO TO 20
    1 J=I+1
      IF (J.EQ.(NSQ+1)) J=1
      K=2
      IF (LS(J,K).GE.0) GO TO 2
      M=M+1
      J1(M)=J
      K1(M)=K
      IS=2
      GO TO 20
    2 J=I-NSIDE
      IF (J.LT.1)  J=J+NSQ
      K=3
      IF (LS(J,K).GE.0) GO TO 3
      M=M+1
      J1(M)=J
      K1(M)=K
      IS=3
      GO TO 20
    3 J=I+NSIDE
      IF (J.GT.NSQ) J=J-NSQ
      K=4
      IF (LS(J,K).GE.0) GO TO 4
      M=M+1
      J1(M)=J
      K1(M)=K
      IS=4
      GO TO 20
    4 IF (M.EQ.0) RETURN
      CALL DRAND(RNUM)
      IF (MF.GT.0) GO TO 5
      M3=INT(RNUM*FLOAT(M))+1
      J=J1(M3)
      K=K1(M3)
      CALL DRAND(RNUM)
      SIGR1=SIGR/FLOAT(M)
      GO TO 6
    5 M3=INT(RNUM*FLOAT(MF))+1
      J=JF(M3)
      K=KF(M3)
      CALL DRAND(RNUM)
      SIGR1=SIGR/FLOAT(MF)
    6 IRETRY=-INT(SIGR1*ALOG(RNUM))+1+NOW
      M4=-LS(J,K)
```

```
          NSAVE=ND(J,M4,1)
          ND(J,M4,1)=IRETRY
          IF (NSAVE.EQ.0) GO TO 218
          NZ=ND(J,M4,3)
S             LH,4        NZ
S             STH,4       IDSUC
S             LI,1        1
S             LH,4        NZ,1
S             STH,4       IOT
          IF (NEX(J,2).EQ.0)  GO TO 219
          IF (M4.EQ.NEX(J,2))  GO TO 219
          MSTOR=NEX(J,2)
          DO 214 M1=1,M
          IF (MSTOR.EQ.0)  GO TO 218
          IF (MSTOR.EQ.M4)  GO TO 217
          MSTOR1=MSTOR
          NZ=ND(J,MSTOR,3)
S             LH,4        NZ
S             STH,4       MSTOR
      214 CONTINUE
          GO TO 218
      219 NEX(J,2)=IDSUC
          IF (IDSUC.EQ.0)  GO TO 220
          NEX(J,1)=ND(J,IDSUC,1)
          GO TO 221
      220 NEX(J,1)=0
      221 ND(J,M4,3)=IOT
          GO TO 218
      217 NZ=ND(J,MSTOR1,3)
S             LI,1        1
S             LH,4        NZ,1
S             LW,5        IDSUC
S             STH,5       4
S             STH,4       NZ
          ND(J,MSTOR1,3)=NZ
          ND(J,M4,3)=IOT
      218 MEX=NEX(J,1)
          IG1=J
          IG2=M4
          CALL ORDERQ
          IF (MEX.EQ.NEX(J,1))  GO TO 230
          IF (J.EQ.INEXT)  GO TO 719
          IF (INEXT.EQ.0)  GO TO 719
          MSTOR=INEXT
          DO 714 M1=1,65
          IF (MSTOR.EQ.0)  GO TO 718
          IF (MSTOR.EQ.J)  GO TO 717
          MSTOR1=MSTOR
      714 MSTOR=NXND(MSTOR)
          GO TO 718
```

```
  719 INEXT=NXNO(J)
      NXNO(J)=0
      GO TO 718
  717 NXND(MSTOR1)=NXND(J)
      NXNO(J)=0
  718 IG1=J
      CALL ORDERN
  230 CONTINUE
      RETURN
   20 IT=-LS(J,K)
      IF (ISEFO.EQ.C) GO TO (1,2,3,4),IS
      NZ=ND(J,IT,2)
S         LI,1       2
S         LB,4       NZ,1
S         STW,4      IDFST
      IF (IOEST.EQ.I) GO TO 21
      GO TO (1,2,3,4), IS
   21 MF=MF+1
      JF(MF)=J
      KF(MF)=K
      GO TO (1,2,3,4), IS
      END
C
C
      SUBROUTINE ORDERO
      COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
     1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
     2   MRAND, KRANO, NOW, JRATE, ND(64,51,3), MESG(64,64)
      COMMON IG1, IG2, IG3, NSIOE, NSQ
      I=IG1
      J=IG2
      IF (ND(I,J,1).EQ.0) RETURN
      NZ=ND(I,J,3)
S         LI,1       1
S         LH,4       NZ,1
S         STW,4      IOUFUF
      MSTOR=NEX(I,2)
      IF (MSTOR.EQ.0) GO TO 12
      IF (ND(I,MSTOR,1).GE.ND(I,J,1)) GO TO 14
      DO 9 M=1,51
      IF (MSTOR.EQ.C) GO TO 11
      IF (ND(I,MSTOR,1).GE.ND(I,J,1)) GO TO 10
      MSTOR1=MSTOR
      NZ=ND(I,MSTOR,3)
S         LH,4       NZ
S         STW,4      MSTOR
    9 CONTINUE
      WRITE (108,21)
   21 FORMAT (1H1,12HERROR-ORDERO)
      WRITE (108,100) I,J,NEX(I,2)
```

```
        DO 101 MF=1,51
        WRITE (108,102) MF,ND(I,MF,1)
        NZ=ND(I,MF,2)
S          LH,8       NZ
S          STH,R      ILF
S          LI,1       2
S          LB,8       NZ,1
S          STH,R      IDE
S          LI,1       3
S          LB,8       NZ,1
S          STH,8      ICE
        WRITE (108,103) ILF, IDE, ICE
        NZ=ND(I,MF,3)
S          LH,8       NZ
S          STH,5      ISS
S          LI,1       1
S          LH,5       NZ,1
S          STH,R      IQQ
    101 WRITE (108,104) ISS,IQQ
    100 FORMAT (2X,3(I3,3X))
    102 FORMAT (5X,I2,3X,I9)
    103 FORMAT (10X,3(I8,2X))
    104 FORMAT (10X,2(I3,2X))
        RETURN
    12  NEX(I,1)=ND(I,J,1)
        NEX(I,2)=J
        ND(I,J,3)=IQUEUF
        RETURN
    11  ND(I,J,3)=IQUEUE
        GO TO 15
    10  CONTINUE
S          LW,4       IQUEUF
S          LW,5       MSTOR
S          STH,5      4
S          STW,4      NZ
        ND(I,J,3)=NZ
    15  NZ=ND(I,MSTOR1,3)
S          LI,1       1
S          LH,4       NZ,1
S          LW,5       J
S          STH,5      4
S          STW,4      NZ
        ND(I,MSTOR1,3)=NZ
        RETURN
    14  NFX(I,1)=ND(I,J,1)
        NFX(I,2)=J
S          LW,4       IQUEUF
S          LW,5       MSTOR
S          STH,5      4
S          STW,4      NZ
```

```
      ND(I,J,3)=N7
      RETURN
      END
C
C
      SUBROUTINE ORDERN
      COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
     1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
     2   MRAND, KRAND, NOW, JRATE, ND(64,51,3), MERG(64,64)
      COMMON IG1, IG2, IG3, NSIDE, NSQ
      I=IG1
      IF (NEX(I,1).EQ.0) RETURN
      MSTOR=INEXT
      IF (INEXT.EQ.0) GO TO 12
      IF (NEX(MSTOR,1).GE.NEX(I,1)) GO TO 14
      DO 9 M=1,65
      IF (MSTOR.EQ.0) GO TO 11
      IF (NEX(MSTOR,1).GE.NEX(I,1)) GO TO 10
      MSTOR1=MSTOR
    9 MSTOR=NXND(MSTOR)
      WRITE (108,21)
   21 FORMAT (1H1,12HERROR-ORDERN)
      RETURN
   12 INEXT=IG1
      NXND(I)=0
      RETURN
   11 NXND(MSTOR1)=IG1
      NXND(I)=0
      RETURN
   10 NXND(MSTOR1)=IG1
      NXND(I)=MSTOR
      RETURN
   14 INEXT=IG1
      NXND(I)=MSTOR
      RETURN
      END
C
C
      SUBROUTINE MESSAG
      COMMON N, INEXT, MULT(6), SIGMA, SIGR, RMU, NS(64), LS(64,5),
     1   NEX(65,2), NXND(65), MESS(4), LSM(64), NRAND(128), LRAND,
     2   MRAND, KRAND, NOW, JRATE, ND(64,51,3), MERG(64,64)
      COMMON IG1, IG2, IG3, NSIDE, NSQ
      I=IG1
      CALL DRAND(RNUM)
      ND(I,51,1)=NOW-INT(SIGMA*ALOG(RNUM))+1
      IF ((NS(I).EQ.0).OR.(LSM(I).NE.0))  RETURN
    1 CALL DRAND(RNUM)
      J=INT(RNUM*FLOAT(NSQ))+1
      IF (I.EQ.J)  GO TO 1
      2   CALL DRAND(RNUM)
          ILENTH=-INT(RMU*ALOG(RNUM))+1
          MESS(2)=J
          MESS(3)=ILENTH
          RETURN
          END
     SYMBOL
                DEF        ISITON
     ISITON     CAL2,1     0
                RD,0       0
                STCF       3
                CAL2,1     1
                SCS,3      4
                LW,4       13
                LW,13      1,4
                AND,3      *13
                B          2,4
                END
```

D. Summary of Relevant Queueing Formulae

In the main body of this work we consider M/M/k queueing systems, i.e., stochastic service systems which experience Markovian arrivals and in which customers depart after receiving an amount of service time that is exponentially distributed and is given by one of k servers. If there are n customers presently in the system, then a customer will arrive in the next instant of time Δt with probability $\lambda_n \Delta t + 0(\Delta t)$ and a customer will depart in the next instant of time with probability $\mu_n \Delta t + 0(\Delta t)$.

The stationary probability of finding n customers in the system is related to $p_0 = P[\text{empty system}]$ in the following way:

$$p_n = p_0 \prod_{i=0}^{n-1} \frac{\lambda_i}{\mu_{i+1}}$$

which is valid for all $n \geq 0$ if we define $\prod_{i=0}^{-1} = 1$. Then p_0 is found from the fact that if this is to be a valid probability distribution

$$\sum_{n=0}^{\infty} p_n = 1$$

If $\lambda_n = \lambda$ and $\mu_n = \mu$ for all n, then

$$p_n = p_0 \prod_{i=0}^{n-1} \frac{\lambda}{\mu} = p_0 \left(\frac{\lambda}{\mu}\right)^n$$

therefore

$$p_n = (1 - \rho)\rho^n \text{ for } \rho < 1 \qquad \text{where } \rho = \frac{\lambda}{\mu}$$

is called the "utilization factor"

$$\rho = 1 - p_0 = P[\text{system is busy}]$$

For an infinite server system every customer has his own server. We take $\lambda_n = \lambda$, $\mu_n = n\mu$, then

$$p_n = p_0 \prod_{i=0}^{n-1} \frac{\lambda}{(i+1)\mu} = \frac{p_0}{n!}(\frac{\lambda}{\mu})^n$$

therefore

$$p_n = \frac{e^{-\lambda/\mu}}{n!} (\frac{\lambda}{\mu})^n$$

and

$$\rho = 1 - p_0 = 1 - e^{-\lambda/\mu}$$

A busy period in a queueing system begins when a customer arrives to an empty system. The busy period continues as long as there is at lease one customer in the system, and the busy period ends the first time that a customer departs leaving behind him an empty system. For the M/M/1 system with $\lambda_n = \lambda$, $\mu_n = \mu$ the probability density of the length t of a busy period is

$$p(t) = i/t\sqrt{p}\, e^{-(\sigma + \mu)t}\, I_1(2t\sqrt{\sigma\mu})$$

where again $p = \frac{\lambda}{\mu}$ and $I_1(x)$ is the modified Bessel function of the first kind, of order one [19]. The average length of the busy period is simply

$$\frac{1}{\mu(1 - \rho)}$$

For an excellent treatment of queueing theory the reader is directed to the book by Cox and Smith [7].